T.C.

İNÖNÜ UNIVERSITY

GRADUATE SCHOOL OF SCIENCE AND TECHNOLOGY

COMPUTER ENGINEERING DEPARTMENT

(Fen Bilimleri Enstitüsü)

(Bilgisayar Mühendisliği Bölümü)

DESIGNING CONTROLLERS FOR PATH PLANNING APPLICATIONS TO MOBILE ROBOTS WITH HEAD-CAMERAS

(Mobil Robotlara Yol Planlama Uygulamaları için Tepe Kameralar ile Kontrolörler Tasarlama)

Emrah DÖNMEZ

D3615190352

PHILOSOPHY OF DOCTORATE (Ph.D.) THESIS

THESIS ADVISOR

Ass. Prof. Dr. A. Fatih KOCAMAZ

MALATYA

KASIM 2018

Tez Başlığı: Designing Controllers for Path Planning Applications to Mobile Robots with Head-Cameras

Tezi Hazırlayan: Emrah DÖNMEZ

Sınav Tarihi: 15 Kasım 2018

Yukarıda adı geçen tez jürimizce değerlendirilerek Bilgisayar Mühendisliği Ana Bilim Dalında Doktora Tezi olarak kabul edilmiştir.

Sınav Jüri Üyeleri:

Prof. Dr. İbrahim TÜRKOĞLU
Fırat Üniversitesi

Prof. Dr. Ali KARCI
İnönü Üniversitesi

Doç. Dr. Bilal ALATAŞ
Fırat Üniversitesi

Doç. Dr. Muhammed Fatih TALU
İnönü Üniversitesi

Tez Danışmanı: **Dr. Öğr. Üyesi Adnan Fatih KOCAMAZ**
İnönü Üniversitesi

İnönü Üniversitesi Fen Bilimleri Enstitüsü Onayı

Prof. Dr. Halil İbrahim ADIGÜZEL
Enstitü Müdürü

HONOR WORD

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Emrah DÖNMEZ

ONUR SÖZÜ

Doktora Tezi olarak sunduğum “Designing Controllers for Path Planning Applications to Mobile Robots with Head-Cameras” / “Mobil Robotlara Yol Planlama Uygulamaları için Tepe Kameralar ile Kontrolörler Tasarlama” – TR başlıklı bu çalışmanın, bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurmaksızın tarafımdan yazıldığını ve yararlandığım bütün kaynakların hem metin içinde hem de kaynakçada yöntemine uygun biçimde gösterilenlerden oluştuğunu belirtir, bunu onurumla doğrularım.

Emrah DÖNMEZ

ABSTRACT

Ph.D. Thesis

DESIGNING CONTROLLERS FOR PATH PLANNING APPLICATIONS TO MOBILE ROBOTS WITH HEAD-CAMERAS

İnönü University
Graduate School of Science
Computer Engineering Department

123 + xv page

2018

Advisor: Ass. Prof. A. Fatih KOCAMAZ

In this thesis study, two different visual based controllers and an adaptive potential field based on path planning methods are designed for a differential drive mobile robot. The designed methods are operated in a multi-camera environment with fixed head camera configuration. Configuration space hosts a number of static obstacles. The controller performs robot motions until a pre-defined target is reached. For each controller two different positioning models are utilized. A weighted graph and a triangle model have been proposed. This study is comprised of three stages. In first stage; a simple go-to-goal controller designed for an obstacle free configuration space. In second stage, designed controller has been fused with a modified path planning method (for obstacle avoidance) and a newly designed controller. In last stage; an expandable configuration space is created with multi-camera device and controllers have been adapted to this new configuration space.

The camera(s) captures image frames in an interior space. A real-time system tracks the configuration space in consecutive frames to detect global positions of a mobile robot, target and obstacles. A graph structure is formed by assuming robot wheels and target as nodes in weighted graph positioning model. Distances between nodes are assigned as weights to the graph edges. A virtual triangle is formed between the robot wheels and target in triangle positioning model. The angles between edges are assigned as interior angles to the triangle corners. Both graph weights and triangle angles are input parameters according to the used positioning model for designed controllers.

In first stage; go-to-goal behavior is modeled for the obstacle free environment. The general Gaussian function is utilized to determine the velocity of wheels in designed controller for both positioning models, separately. We compare outputs of controller with several conventional methods which are PID and Fuzzy-PID. Then it has been seen that the mobile robot control has been performed with high precision and accuracy by employing the developed visual-based Gaussian controller.

In second stage; a decision tree based mobile robot control and an adaptive potential field-based obstacle avoidance control have been developed for a static obstacle hosted environment. Then, we harmonized both control unit and performed a real-world experiment. Firstly, a path plan extracted by using adaptive potential field method. To calculate potentials virtual range sensors are used. Secondly, decision tree-based controller has advanced the wheeled mobile robot (WMR) on this reference trajectory path in real-time. Experimental environment has included static obstacles and different configuration spaces. Efficiency and robustness of potential field method has greatly improved by utilizing optimal parameters found with adaptive potential field design. We have acquired and evaluated both simulation and real-world experiment data from control process.

Finally, all the designed controllers and models have been combined and a new control infrastructure has been developed to work with multi-camera device configuration in third stage. We proposed a new multi-camera operating model by stitching multi-images into one image. Developed path planning and path dividing methods are implemented on this stitched image. Experimental results show designed controllers and methods successfully characterize WMR motions for multi-camera model under different configuration spaces.

***Keywords*:** Visual based control, Path planning, Gaussian controller, Decision tree controller, Artificial potential field

ÖZET

Doktora Tezi

MOBİL ROBOTLARA YOL PLANLAMA UYGULAMALARI İÇİN TEPE KAMERALAR İLE KONTROLÖRLER TASARLAMA

İnönü Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği
Yazılım Anabilim Dalı

123 + xv sayfa

2018

Danışman: Dr. Öğr. Üyesi A. Fatih KOCAMAZ

Bu tez çalışmasında, diferansiyel tahrikli bir gezgin robot için iki farklı görü tabanlı kontrolör ve potansiyel alan yöntemine dayalı uyarlamalı yol planlama metodu tasarlanmıştır. Tasarlanan metotlar çoklu-kamera ortamında sabit tepe kamera konfigürasyonu ile çalıştırılmıştır. Konfigürasyon uzayı birden fazla statik engel barındırmaktadır. Kontrolör ön-tanımlı bir hedefe ulaşıncaya kadar robot hareketlerini yürütmektedir. Her bir kontrolör için iki farklı pozisyonlama yönteminden faydalanılmıştır. Bu kapsamda; bir ağırlıklı çizge ve bir de trigonometrik üçgen modelleri önerilmiştir. Bu tez çalışması üç aşamadan oluşmaktadır. İlk aşamada; engel içermeyen bir konfigürasyon uzayı için temel bir hedefe-gitme kontrolörü tasarlanmıştır. İkinci aşamada; yeni olarak tasarlanan bir hedefe-gitme kontrolörü ile yeni olarak tasarlanan bir hedeften kaçınma kontrolörü kaynaştırılmıştır. Son aşamada ise; çoklu-kamera cihazları ile genişletilebilir bir konfigürasyon uzayı oluşturulmuştur ve kontrolörler bu yeni konfigürasyon uzayına uyarlanmıştır.

Kamera(lar) bir iç mekânda imge çerçevelerini yakalarlar. Robot, hedef ve engellerin global konumlarını tespit etmek amacıyla; konfigürasyon uzayı ardışık çerçevelerde gerçek zamanlı olarak izlenmektedir. Ağırlıklı çizge konumlandırma modelinde robot tekerleri ve hedef birer düğüm varsayılarak bir çizge yapısı oluşturulur. Düğümler arasındaki mesafe değerleri çizge kenarlarına ağırlık olarak atanmaktadır. Üçgen konumlandırma modelinde robot tekerleri ve hedef arasında sanal bir üçgen yapısı oluşturulur. Üçgenin kenarları arasındaki iç açılar üçgen köşelerine açı değerleri olarak atanmaktadır. Hem çizge ağırlıkları hem de üçgen iç açıları kullanılan konumlandırma modeline göre tasarlanmış kontrolörler için giriş parametreleri olarak kullanılmaktadır.

İlk aşamada; engel içermeyen bir ortam için hedefe-gitme davranışı modellemiştir. Gaussian fonksiyonu her iki konumlandırma modeli için teker hız değerlerini belirlemek amacıyla varsayılan kontrolör içerisinde kullanılmıştır. Bu kontrolörden elde edilen çıktılar ise iki geleneksel kontrol yöntemleri olan PID ve Fuzzy-PID ile karşılaştırılmıştır. Tasarlanan görü tabanlı Gaussian kontrolörü kullanarak mobil robot kontrolünün yüksek hassasiyet ve doğruluk ile gerçekleştirildiği görülmüştür.

İkinci aşamada; statik bir ortam için karar ağacı tabanlı bir gezgin robot kontrolü ve uyarlanabilir potansiyel alan tabanlı engel kaçınma kontrolü geliştirilmiştir. Daha sonra, her iki kontrol birimi uyumlu hale getirilmiş ve gerçek bir dünya deneyi gerçekleştirilmiştir. İlk olarak, uyarlamalı potansiyel alan yöntemi kullanarak bir yol planı çıkartılmıştır. İkinci olarak, karar ağacı tabanlı kontrolör tekerlekli gezgin robotu (TMR) bu referans yörünge yolu üzerinde ilerletmeye başlamıştır. Deneysel ortam statik engeller ve farklı konfigürasyon uzayları içermektedir. Uyarlamalı potansiyel alan yöntemi ile bulunan optimum parametrelerden yararlanarak potansiyel alan yönteminin verimi ve dayanıklılığı büyük ölçüde iyileştirilmiştir. Kontrol işleminden simülasyon ve gerçek dünya deneysel verileri elde edilmiş ve değerlendirilmiştir.

Nihai olan üçüncü aşamada ise tasarlanan tüm kontrolörler ve modeller birleştirilmiş ve çoklu-kamera cihaz konfigürasyonu ile çalışabilecek şekilde yeni bir kontrol altyapısı geliştirilmiştir. Çok görüntüyü dikişleme yöntemiyle tek bir görüntüde birleştirerek yeni bir çoklu kamera işletim modeli önerilmiştir. Bu dikişli görüntü üzerinde geliştirilen yol planlama ve yol bölütleme yöntemleri uygulanmıştır. Deneysel sonuçlar, tasarlanan kontrolörlerin çoklu kamera konfigürasyonu için de TMR hareketlerini farklı konfigürasyon uzayları başarılı bir şekilde karakterize ettiğini göstermiştir.

Anahtar Kelimeler: Görü tabanlı kontrol, Yol planlama, Gaussian kontrolör, Karar ağacı kontrolör, Yapay potansiyel alan

PREFACE

This dissertation is submitted for the degree of Doctor of Philosophy at the Inonu University. The research defined herein has been implemented under the supervision of Assistant Professor A. Fatih Kocamaz in the Computer Engineering Department, Faculty of Engineering at Inonu University, between September 2015 and October 2018.

Part of this work has been presented in the following publications/projects:

TÜBİTAK 1002 Hızlı Destek Projesi (Quick Support Project) Project No: 116E568 Completed (2018)

Dönmez E. and Kocamaz A. F., " Multi Target Task Distribution and Path Planning for Multi-Agents," 2018 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, 2018, pp. 1-7.

Dönmez E., A. Kocamaz F., and Dirik M., "A Vision-Based Real-Time Mobile Robot Controller Design Based on Gaussian Function for Indoor Environment", Arab. J. Sci. Eng., (2017) 1–16.

Dönmez E., Kocamaz A. F. and Dirik M., "Bi-RRT path extraction and curve fitting smooth with visual based configuration space mapping," 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, 2017, pp. 1-5.

Dirik M., Kocamaz A. F. and **Dönmez E.**, "Visual servoing based path planning for wheeled mobile robot in obstacle environments," 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, 2017, pp. 1-5.

Dönmez E., Kocamaz A. F. and Dirik M., "Visual based path planning with adaptive artificial potential field," 2017 25th Signal Processing and Communications Applications Conference (SIU), Antalya, Turkey, 2017, pp. 1-4.

Dirik M., Kocamaz A. F. and **Dönmez E.**, "Static path planning based on visual servoing via fuzzy logic," 2017 25th Signal Processing and Communications Applications Conference (SIU), Antalya, Turkey, 2017, pp. 1-4.

Dönmez E., Kocamaz A. F., Dirik M. "Robot Control with Graph Based Edge Measure in Real Time Image Frames". 24th Signal Processing and Communication Application Conference (SIU), pp. 1789-1792., 2016

Dirik M., Kocamaz A. F., **Dönmez E.**, "Vision-Based Decision Tree Controller Design Method Sensorless Application by Using Angle Knowledge". 24th Signal Processing and Communication Application Conference (SIU), pp. 1849-1852., 2016

Dönmez E., Kocamaz A. F., Dirik M. "Robotic Positioning Method Design through Image Based Virtual Path with Multi-Head Camera Infrastructure". International Conference on Natural Science and Engineering (Icnase'16), pp. 2278-2285., 2016

Kocamaz A. F., Dirik M., **Dönmez E.** "Head Camera-Based Nearest Neighborhood Relations Algorithm Optimization and the Application of Collecting the Ping-Pong Ball". International Conference on Natural Science and Engineering (Icnase'16), 2385-2390., 2016

ACKNOWLEDGEMENTS

I am extremely grateful to my supervisor **Professor *A. Fatih KOCAMAZ*** for his endless support, enthusiasm, knowledge and friendship. I would like to thank Professors; *M. Fatih TALU*, *B. Baykant ALAGÖZ* and *Ali KARCI* for their supports and the provision of the laboratory facilities in the Department of Computer Engineering at Inonu University. I also thank M.Sc. *Yahya ALTUNTAŞ* and Dr. *Nuh ALPASLAN* for their valuable comments.

I am indebted to Inonu University Robotic Lab and researchers for giving comments about this thesis study, and in particular to Researcher *Mahmut DİRİK* for his valuable comments and feedbacks throughout the thesis.

Thanks also to my friends and to the people I met at Inonu University, other Universities, Companies and Institutes at my time in Inonu.

Finally, I take this opportunity to express my gratitude to ***my family*** for their love, unfailing encouragement and support.

TABLE OF CONTENTS

LIST OF ABBREVATIONS

PID	=	Proportional, Integral, Derivative
PI	=	Proportional, Integral
VBC	=	Visual Based Control
RRT	=	Rapidly Exploring Random Tree
BFS	=	Breadth First Search
DFS	=	Depth First Search
APF	=	Artificial Potential Field
WMR	=	Wheeled Mobile Robot
CCD	=	Charge Coupled Device
CAD	=	Computer-Aided Design
RFID	=	Radio Frequency Identification
GPS	=	Global Positioning System
NN	=	Nearest Neighbor
MLC	=	Monte Carlo Localization
SLAM	=	Simultaneous Localization and Mapping
ISS	=	Input-to-State Stability
CPP	=	Coverage Path Planning
BVP PP	=	Boundary Value Problem
LOPF	=	Locally Oriented Potential Field
GSA	=	Gravitational Search Algorithm
PSO	=	Particle Swarm Optimization
MR	=	Mobile Robot
BPF	=	Bacterial Potential Field
BEA	=	Bacterial Evolutionary Algorithm
Wi-Fi	=	Wireless Fidelity
CPU	=	Central Processing Unit
RPM	=	Revolution per Minute
HDD	=	Hard Disk
2D	=	Two Dimensional
RGB	=	Red-Green-Blue
HSV	=	Hue-Saturation-Value
LF	=	Local Frame
EDV	=	Euclidean Distance Value
A-APF	=	Adaptive Artificial Potential Field
DT	=	Decision Tree
NRC	=	Next Range Condition
GPU	=	Graphical Processing Unit
Bi-RRT	=	Bidirectional Rapidly Exploring Random Trees
D-RRT	=	Dynamic Rapidly Exploring Random Trees
RRTCAP	=	Rapidly Exploring Random Trees Controller and Planner
DTM	=	Discrete Time Motion Model
LAR	=	Least Absolute Residuals
TRA	=	Trust Region Algorithm
LBS	=	Load Balancing System

LIST OF FIGURES

LIST OF TABLES

Intentionally left blank.

1. INTRODUCTION

The control process is a challenging topic in robotics. There are significant number of researches mostly concerning with in-device sensor control with conventional methods which are PID, Fuzzy Control, Fuzzy PI, heuristics etc. [1], [2], [3], [4], [5]. The control process is mostly implemented by using global position and directional angular data with the controller functions [6], [7]. The general controller tasks are executed with data obtained from in-device (or internal) sensors such as encoders, gyroscope, accelerometer and out-device (or external) sensors such as infrared, thermal camera, proximity sensors. By utilizing several sensor information, the angular data are computed and input parameters controller functions are updated to create next robot motion.

Minimizing the error is a critical issue in robotic control for industrial and non-industrial robotics. There are two types of errors in robotic systems; non-systematic and systematic errors. Non-systematic errors are generally caused by falling, hitting, and sliding etc. On the other hand, systematic errors are generally caused by the erroneous sensor data, encoder and physical form of the robot parts. The general purpose of control methods is compensating these errors until the assigned task(s) are completed [8].

Visual based control (VBC) systems are used to model a dynamic or static system by employing visual features obtained from images provided by camera(s) [9], [10], [11]. It can be said that a robot controlling process can be modeled by using an image perception system. This process is performed by analyzing each of the image frames obtained through the imaging sensor. Similarly, to conventional controllers, the aim of VBCs is eliminating errors and decreasing cost of the motion to an acceptable level. The main benefits of the VBC (or visual servoing) are that it requires small amount of data from sensor(s), appropriate to control multiple agents and internal or external sensors on the robots usually are not required. In terms of expandability of configuration space, it ensures more working field by increasing number of imaging device(s).

Visual servoing is broadly implemented in robotic researches. In early researches, controllers for robotic arm manipulators have been modeled by using Jacobean-based methods and visual features generally with eye-in-hand configuration [12], [13]. In

later researches, control tasks for humanoid robots, mobile robots, autonomous (or self-driving) vehicles etc. have been carried out by image-based visual controllers [14], [15], [16], [17]. In recent researches, the real time robotic systems, multitasking robotics and unmanned aerial vehicles have been developed by utilizing mostly estimation-based methods with image sensor hardware [18], [19], [20].

There are two camera configurations used in most of the VBC studies; firstly, the camera(s) can be equipped onto the robot with eye-in-device configuration. Robot determines its global position according to the object detection, measured depth information from the images and distance data from the encoder values. Secondly, camera can be equipped to a fixed position with eye-out-device configuration. Robot determines its global position according to only the measured distance information calculated on the images. Additional data like encoder values can be used in this configuration as well. In both configurations, WMR control procedure highly depends on the processing acquired images from the cameras and extract information about the resided environment. Therefore, compared to the classical robot control methods, it can be said that VBC is next phase of robot control models. Because information about environment are obtained not only with sensors (range, altitude, balance etc.) but also with imaging sensors. This study focuses on eye-out-device camera configuration to control a WMR. Locomotion of a mobile robot with eye-out-device configuration resembles that a child plays with his wheeled toy car with his hands by looking car from the above.

Whether a controller infrastructure is built on visual or non-visual data, the primary issues are accuracy, robustness and speed of the methods [7]. A general control model should provide both speed and accuracy to enhance robustness. A simple control model having low complexity is a good option for real time applications. Therefore, the design of controller chunks and complexity change according to the aim of application. E.g. a service robot requires less precision compare to a surgical robot which requires high precision. In addition, the configuration space contains several factors like ground form, light level, friction coefficient, humidity, temperature, atmospheric pressure etc. [6]. A control system can be non-sensitive or sensitive to these factors according to the specifications in the working environment.

According to the specified tasks, a mobile robot controller is generally created as module(s)/chunk(s). Main tasks for a mobile robot are;

I. Reaching to an unknown or a specific target,

II. Tracking a pre-defined path plan,

III. Static and dynamic obstacle(s) avoidance

The first introduced task “reaching a target” is known as go-to-goal behavior. This behavior is modeled as the most basic control module of a mobile robot. The second mentioned task “tracking a trajectory” corresponds to following a detected/pre-defined path. This behavior can be considered as a collection of go-to-goal behaviors rather than go to a single target position. In other words, a trajectory actually consists a series of points and each point in this trajectory can be considered as a target point. Therefore, it can be said that this behavior is sequential iteration of a group go-to-goal control tasks. The last pointed out task “avoiding obstacles” refers the performing go-to-goal behavior without crashing to any object. This means that go-to-goal and obstacle avoidance behaviors are fused to perform a given task. In a static environment, obstacle avoidance can be performed with two methods. In first method; since obstacles are static, a path plan can be extracted from environment before starting the motion process of robot. After creating a path plan mobile robot simply tracks this path until reaching to the target position. In other words, go-to-goal behavior is converted to a trajectory tracking behavior. In second method; the mobile robot is starting with simply go-to-goal behavior. When an obstacle is detected on the path, motion behavior is changed to obstacle avoidance from go-to-goal and mobile robot simply avoids obstacle with minimum additional movements. In a dynamic environment if a robot has to perform a continuous locomotion, there is only one option; the robot should perform go-to-goal behavior and instant obstacle avoidance behavior together. Because the dynamic obstacles (other robots, humans, vehicles etc.) can appear on the path in any time. So, we can’t model it with a trajectory tracking behavior after an offline path extraction. If continuous locomotion is not an issue, then robot can be stopped when a dynamic obstacle appears on the path. After this obstacle leaves from the path robot is triggered to continue its previous motion model. Except from these three main tasks, there can be sub-tasks and other environment specific tasks. For example; an autonomous car tracking only

specific targets such as traffic signs and make a movement decision with respect to meaning of these signs.

Path planning is one of the basic components of the robot control process. It simply concerns modelling a path between an initial position configuration and a final position configuration. Path plan have to be extracted from an operating environment by considering obstacles (wall, door, any object etc.). The key elements in path planning are admissible path cost (or efficiency), path safety and robustness [21]. Path planning is made according to the problem structure. There are two approaches; global and local used to extract a path plan from a given environment or configuration space [22]. Global approaches are divided into two categories; retraction methods and decomposition methods. The retraction methods recursively reduce the initial problem dimension by considering sub-part of the configuration space. Decomposition methods characterize the obstacle free regions of a given configuration space. On the other hand, local approaches are mainly use the distance parameter to the target while avoiding obstacles in motion state. This distance value (gradient of the cost function) is generally guide the local method. Local approaches are more efficient to overcome the complex robots. Moreover, path planning can be made with randomized methods or stochastic which considers building a graph and find a local minimum at each iteration. In addition to configuration space, path planning can be considered in trajectory space. In this space a straight line is created between initial and final configuration while all of the obstacles in the environment are neglected. In the next step this path progressively reshaped by reducing improper parts (e.g. intersecting with an obstacle) of the acquired path.

There are a number of commonly known path planning methods. Each method aims to find a convenient path with minimum cost. Each method aims to find a convenient path with minimum cost. Dijkstra method [23], [24] finds shortest path between given two points (nodes). A* [25] is a heuristic contributed version of Dijkstra method. It uses an additional cost function that estimates cheapest path from actual node to the target node in each step. D* [26] is dynamic version of A*. In an unknown environment, method starting to work like A*, when a previously unknown obstacle is detected, this information is added to map as a new map information. Then, if it is necessary shortest path is updated according to this new map. There are several common methods using random branching by starting a defined initial point

to a target point. The rapidly exploring random tree (RRT) [27] aims to find a path between predefined start and finish coordinates in an unknown configuration space. As its name emphasize it starts by branching randomly without crashing an obstacle at each iteration until reaching to the final/desired position. The other type of RRT method is Bi-RRT (Bidirectional-RRT) [28], it starts to branch from both starting and finishing positions. Two trees approximate to each other in each iteration step. The method stops the searching process, when any two branches of trees intersect at an undefined position. To search a given position tree based BFS (Breadth First Search) [29] and DFS (Depth First Search) [30] searching methods are also widely used in a remarkable number of studies. There are probabilistic and statistical path planning methods which are used learning-based methods.

Except from graph-based methods a path planning can be extracted from working space by using a potential field inspired method which is known as APF – Artificial Potential Fields. APF is firstly introduced by [31], it is a commonly used method to create a path plan between an initial position and final position in an obstacle-hosted environment. If potential field is considered as electrical field, then robot and obstacles have same charges and target has opposite charge. Main idea is based that robot configuration is treated as an electron which is attracted by target and repulsed by obstacles. The resulting trajectory of this electron (robot) is the obstacle free path in configuration space. However, there are several problems which causes to electron (robot) can be trapped in a local minima or unstable oscillation for pure APF.

In this study, we design and experiment a Gaussian and decision tree inspired WMR go-to-goal behavior controllers based on visual servoing by using two simple graph or triangle positioning models separately. Then, we created a path planning design with adaptive potential field approach and developed the decision tree based visual controller to navigate the robot on this formed path. Lastly, a multi-camera configured environment is utilized as a testbed. A load balancing system developed for multi target and multi robot model as an additional experiment. The designed methods have experimented with several configuration spaces. The designed methods work with great accuracy and speed. In section 2 we touch existing studies. Problem definition is given in section 3. We focus on theories, materials and methods in section 4. Test results are demonstrated in section 5. Ultimately, conclusion and future works are handled in section 6.

2. RELATED LITERATURE WORKS

2.1. Visual Based Control (VBC) Studies

Ziaei et al [32] form a global path plan for an omni-directional mobile robot by utilizing a single CCD imaging device. To avoid obstacles, the APF (Artificial Potential Field) method is executed in the configuration space. The obstacles are static objects and borders of image obtained from camera. The robot 3D CAD model is used to perform kinematic control assignments to physical servo interface. **Johnson et al** [33] proposed a multi-robot model to detect position of mobile robots by tracking color LEDs on the robots from a camera, simultaneously. They underline that the false positive LED lights are an explicit challenging in the system. **Chen and Lee** [34] have performed calibration of a fish-eye camera. The camera captures images of the configuration space to implement a visual servoing control. The obstacles detection is performed with image processing. A rectangle is used to frame the objects. If there are no intersections in the image, then each corner of these rectangles is connected. Eventually a connected graph is formed. The Dijkstra method is used to discover a shortest and safe path to the target position in the graph. It is said that the dilation of the detected objects cause to losing of safe paths. **Mezouar and Chaumette** [35] have studied on a visual-based feedback control system. It is claimed that an image-based control and a path plan extracted from image space are pieced together. They said that proposed feedback control system demonstrates robustness against the modeling errors. The robot has been tracked with a single camera and errors are calculated in formed path trajectories. The camera calibration and irregular shape information have been defined as main problems in their study. **Breitenmoser et al** [36] have introduced a localization method for robot-robot systems. It is aimed to obtain relative position of tracked robot in 3D configuration space. In their proposed system A regression-based estimation method is used to model position. **Bista et al** [37] have developed an visual-based method for navigation by employing line segmentations for indoor applications. They use only the 2D image information acquired from an internal imaging device on the robot. It is said that accurate localization and mapping processes are not required for a well-structured navigation system. **Bateux and Marchand** [38] have proposed a visual servoing system based on histogram for indoor/outdoor environments. In their study, the visual features are the histograms.

They said that without disturbing control laws, their proposed system can be applied on any kind of histograms. **Espiau et al** [39] introduced a novel method to vision-based control systems. The vision system is considered as a specific sensor assigned to a task and included in a servo control loop in their basic motive. They defined two key issues in vision-based control task, designing the efficient controllers and performing the definition-specification. **Pauli** [40] investigated learning based robot vision with details. He emphasized that it is desired to develop new-generation robots demonstrating higher degrees of autonomy for fulfilling high-level purposeful tasks in natural and dynamic environments. **Zhao et al** [41] performed a review on image-based control methods used for agricultural robots in harvesting process. They explored object detection in tree canopies and picking objects utilizing visual data as major visual-based control methods and potential applications of these methods in vegetable/fruit harvesting robots. They specified object recognition and coordination of eye-hand as the most significant key issues. **Donmez et al** [42] performed an visual-based control for a mobile robot. They used graph-based velocity control with a basic branching algorithm design. **Dirik et al** [43] implemented a visual-based WMR control in real time acquired images, similarly. It is claimed that the path tracking error is reduced to a smaller value in each control loop.

In addition to these works, there are different control systems types that based on RFID devices to identify real position of the mobile robot. The RFID devices placed to a number of position in configuration space. Similar to the GPS (Global Positioning System) infrastructure by utilizing signal values of RFID, the robot position is identified. Several other researches place the camera on mobile robot vertically, so that the imaging device detects the ceiling surface. This surface of the ceiling is covered with physical markers like shapes, colors etc., then the robot position is detected by using the markers, **Martinelli** [44]. **Elsheikh et al** [45], designed a real-time path planner and navigation method for a non-holonomic mobile robot depend on visual based control. Multi-Stencils Fast Marching as being first part is used to obtain path plan. It is said that if the acquired path plans of fast marching are directly utilized, then safe and smooth path is not guaranteed. **Wang et al** [46], touches upon the adaptive visual-based control for a robotic manipulator. The manipulator is placed under an uncalibrated eye-in-device form with uncertain actuator backlash. It is said that the actuator backlash constraint for control to visual-

based manipulator is not to considered in existing methods. This constraint is inevitable for the robot and effect the dynamic performance, remarkably. **Zhang et al** [47], developed a monocular visual control approach for non-holonomic mobile robots. It is said that the presented method operates well even with both unknown extrinsic camera-to-robot and unknown depth parameters. It is said that the stabilization problem is a challenging issue and still unsolved. They claimed that by utilizing adaptive control and back-stepping method a novel two-stage controller is developed.

2.2. Fundamental Path Planning Studies

Elfes [48] introduced a sonar based real-world mapping and navigation system. An autonomous mobile robot was operated in unknown and unstructured environment. The introduced system utilized sonar range data to establish a multileveled description of the robot's surroundings. It has been said that practical real-world stereo vision navigation systems simply form sparse depth maps of their surroundings. He claimed that the proposed system ensures a sufficiently rich definition of the robot's environment to invoke for more complicated tasks. **Elfes** [49] reviewed occupancy grids which utilizes a probabilistic tessellated representation of spatial information for perception of robot and world modeling, as a new approach. In the real-world experiments, obstacle avoidance was ensured by using potential fields and A* search algorithm. It is claimed that the occupancy grid infrastructure assures a robust and combined approach to a variety of issues in spatial robot perception and navigation. **Borenstein and Koren** [50] designed a new approach titled as virtual force field which associates accuracy grids for obstacle representation and potential fields for navigation. The robot avoided traps like dead-ends or 'U' shaped obstacles by using wall following mechanism in their study. They claimed that their navigation algorithm also takes cognizance of the dynamic behavior of a fast-mobile robot and overcomes the local minimum problem. By inspiring their previous work, **Borenstein and Koren** [51] introduced a new approach referred as vector field histogram that provides the detection of unknown obstacles and avoids collisions while simultaneously steering the mobile robot toward the target. The method utilizes Cartesian histogram grid as a 2D world model. This world model is updated continually with distance data sampled by internal distance sensors. They claimed that vector field histogram method is computationally

efficient, robust, and eliminates misreading(s). It was said that it permits continuous and fast motion of, the mobile robot without stopping for the obstacles. **Murray and Sastry** [52] investigated methods for steering forms with nonholonomic limitations between various configurations. They derived suboptimal trajectories which are not in canonical form. A class of systems which steerable using sinusoids were described in the study. They claimed that building of a trajectory for systems with drift is still an explicit problem. **Laumond et al** [53] presented a fast and precise path planning method based on recursive subdivision of a obstacle-free path produced by a geometric planner neglecting the limitations of motion for their mobile robot. They claimed that the acquired trajectory is improved to yield a path which is of near minimal length in its homotopy class. It is emphasized that the existence of an obstacle-free trajectory is formed by an open connected domain of the acceptable configuration space. **Fierro and Lewis** [54] introduced a controller which provides the combination of a neural network (NN) computed-torque controller and a kinematic controller for the nonholonomic mobile robots. Stability in control is provided by using Lyapunov theory. They claimed that their method does not need information about the cart dynamics generated utilizing an NN back-stepping method. It is said that an NN dynamic controller and a fine-designed kinematic controller may improve the performance of the mobile robot remarkably. **Dellaert et al** [55] introduced a robot localization method with the Monte Carlo Localization (MLC) method, where density of the probability included by maintaining a group of instances which are randomly taken from it is presented. It is said that by employing a sampling-based representation a localization method that can exemplify arbitrary distributions was acquired. They defined sample impoverishment: in the resampling stage, high weighted samples will be chosen multiple times, resulting in a loss of 'variety' as a major problem. **Kuffner and Lavalle** [56] introduced an efficient and simple randomized algorithm for overcoming single-query problems of the path planning in high-dimensional configuration spaces. Their method operates by progressively forming two Rapidly-exploring Random Trees (RRTs) rooted at the initial and the target positions. They defined several performance issues to improve RRT even further. **Cosio and Castaneda** [57] is introduced a novel layout for a mobile robot autonomous navigation, depending on genetic algorithm and artificial potential fields. Genetic algorithm is responsible for automatically determining the specifications of the

optimal potential field. Intermediate targets have been utilized to guide the robot through corridor corners and the door of the suitable room in their method. **Whyte and Bailey** [15], [58] provided a broad introduction to Simultaneous Localization and Mapping (SLAM) problem. They presented the structure of the SLAM problem in standard Bayesian form, and clarifies the SLAM process evolution. They touched several unresolved issues especially for unstructured and dynamic environments.

2.3. Recent Path Planning Studies

Xu et al [59] discussed the commonly known potential field approach for obstacle avoidance in the scope of mobile robots. They indicate the requirement of applying a motion planner for nonholonomic robots and recommend some additions to other potential field-based models to deal with the limitations of car-type robots. The curvature and point mass limitations from car-like mobile robots explained in detail as practical constraints. **Kovacs et al** [60] presented a scheme for mobile robot path planning task in household environments. They extended conventional artificial potential field (APF) method by inspiring motion characteristics of household animals. Mobile robot behaviors are modeled according to possible animal attributes for path planning and goal is assumed as owner or a meal. Actually, they combine APF and Bug Algorithm. It is claimed that by modelling natural motion attributes of animals into the robot, the human–robot interaction transforms to much more natural and intuitive. **Guerra et al** [61] introduced a new method to overcome local minima which occurs in potential field method. It is said that unsteady equilibriums are evaded capitalizing on the built Input-to-State Stability (ISS). Although the robot controlled on extracted trajectory with success, oscillations have emerged while moving in narrow passages. **Jia et al** [62] presented a novel coverage path planning (CPP) algorithm for autonomous exploration robots. The proposed algorithm decomposes the region of interest into cells by discovering landmarks in the environment. Each cell is covered utilizing a zig-zag motion pattern. They claimed that the developed landmark detection is robust to incomplete perception and can be applied to any random shape obstacles. **Bennet and Mclnnes** [63] considered pattern formation and re-configurability in a multi-agent strategy employing a new control method developed over bifurcating potential fields. They claim that the various patterns can be accomplished autonomously through a simple free parameter exchange. It is said that APF method is suitable to implement multi-robot systems.

Romero et al [64] presented an extension for the boundary value problem path planner (BVP PP) to manage multiple robots in a soccer activity. This extension is known as Locally Oriented Potential Field (LOPF) and computes a potential field from the numerical solution of a BVP utilizing local relaxations in different parts of the solution space. This process ensures to manage multiple robots simultaneously, where each robot has different behaviors. **Yan and Li** [65] proposed a fuzzy logic and filter smoothing by using the data from the laser scanning sensor. It is claimed that in a dynamic environment, this algorithm can automatically extract the best path according to the position and size of gaps between the obstacles. They said that fuzzy algorithm and filter smoothing are appropriate to real time systems because of their simplicity and fast response. **Das et al** [66] proposed a novel approach to improve the path plan for multi-agents utilizing gravitational search algorithm (GSA) for a dynamic configuration space. GSA has been improved based on memory data and cognitive parameter of PSO (particle swarm optimization). The algorithm finds obstacle free optimal path from predefined starting position to finishing position for each robot in the environment. It is emphasized that both the obstacles and environment are static relative to the robots. **Montiel et al** [67] introduced a new method that computed optimum paths in environments including dynamic and static obstacles with a WMR for path planning task. The developed method called as Bacterial Potential Field (BPF) and they claimed that it provides an optimal, feasible and safe path. They utilize from both Bacterial Evolutionary Algorithm (BEA) and Artificial Potential Field (APF) to acquire an enhanced flexible path planning method. They emphasize that method takes all the benefits of APF method and reduce its deficiencies. BPF utilizes a WMR model that is generic but realistic. This model takes into account the physical size of WMR and direction in the plane. **Santos et al** [68] proposed a short-term path planner approach for self-driven sailboats which has capability of dealing with upwind situations. In order to achieve this, an initial path is geometrically formed and an optimization is performed over this path, utilizing genetic algorithm. It is claimed that when compared to the brute force approach, the optimization of the model is able to generate similar or better results. **Tan et al** [69] presented an efficient fusion algorithm for the rotary-wing flying robot for solving path planning problem in the 3D mountain environment. This fusion algorithm integrates A* algorithm with artificial potential field method. Both methods improved and optimized for 3D environment. It is emphasized that APF

algorithm is used to smooth the trajectory to improve the performance of path smoothness and traceability. **Donmez et al** [70] have proposed an adaptive artificial path planning (A-APF) method to extract path plan in an obstacle hosted environment. They conducted experiments in real-time. They claimed that A-APF method provides feasible and fast solutions compare to default APF method. **Dirik et al** [71] have proposed a fuzzy-logic decision cluster method for path planning in visual based control systems. They extract fuzzy rules and implemented simulation tests. It is said that the presented method enhances accurate and sensitive mobile robot control procedure. **Donmez et al** [72] have proposed curve smoothing methods on bidirectional rapidly grown random tree (Bi-RRT) path planning method. They used Polynomial, Fourier and Gaussian curve smoothing methods with LAR (Least Absolute Residuals) and Bi-Square weights to reduce path errors. They claimed that the curve smoothing increases the path safety and decreases the path errors and cost, significantly. **Dirik et al** [73] have developed a visual based control system with fuzzy-PID method and creates rule table. They claimed that proposed method is specialized for visual based systems and provides fast and accurate mobile robot control process.

Kamarry et al [74] introduced a novel method to increase the distribution of the nodes in the RRT. This approach enables a compact representation of the working environment by decreasing the nodes redundancy. It is said that the presented biasing method has low computational cost and it is easy to apply. **Kunwook et al** [75] offer an efficient RRT* path planner for hyper-redundant in-pipe robot. They use sliding windows for random sampling in the configuration space to acquire pipeline topology advantages. It is said that the presented method explores applicable paths more efficiently than the conventional methods. **Shan et al** [76] proposed an improved D-RRT (Dynamic – Rapidly exploring Random Trees) path planning method for the application of ALV (Automatic Land Vehicle) working in a dynamic environment. It is said that nonholonomic constraints of the vehicle are combined with three order B-Spline based functions. They claimed that the algorithm guarantees the trackability of the path at the same time. **Melchior and Simmons** [77] defines a novel modification to the default RRT path planning method. The Particle RRT method explicitly consider uncertainty in its working space, similar to the executing of a particle filter. Each extension to the search tree is behaved as a

stochastic process and these extensions are simulated several times. **Heb et al** [78] proposed a trajectory planning approach named as the RRT Controller and Planner (RRTCAP*), assembling the planning stage of a RRT*-based algorithm with the application stage on a physical robot. It is claimed that the presented RRTCAP* is superior to the conventional RRT and RRT*-based path planning methods. **Muñoz et al** [79] introduces two contributions: a mathematical formulation for any DTM that can be used by heuristic search algorithms, and a path planning approach that generates candidate paths which is safer than the ones obtained by previous methods. Developed algorithm, named 3Dana, and it is claimed that the method considers distinct parameters to enhance the quality of path: the maximum permitted slope by the robot and the direction changes through the path tracking procedure.

2.4. Multi-Camera Studies

Visual based robot control is commonly researched in significant number of studies. The main focusing point in these studies are generally; decreasing errors and increasing speed and robustness. There are a number of configurations to implement a visual based robot control infrastructure. Multi-camera configuration is one of them. **Malis et al** [80] have expanded the conventional visual-based control methods to the utilize of multiple imaging devices tracking several segment of an object. The visual based control with the multi-camera has been developed as a chunk of the task cluster method. They claimed that the specific selection of the task module permits them to facilitate the control design and the stability analysis. **Lippello et al** [81] has presented visual servoing method with position information utilizing a mixed eye-in-device multi-camera infrastructure in their study. It is claimed that depending on a modified Kalman filter. This method utilizes the information ensured by all the imaging device without “a priori” distinction, permitting real-time estimation of object position. **Qiu et al** [82] have proposed a robot visual servoing system using multi-camera configuration. The designed system uses switching the vision system between eye-in-device camera and the stereo cameras with voting process. They claimed that the multi-camera infrastructure enable process in a more comprehensive variety of situation than that of either eye-in-hand or stereo camera single configuration. **Yoshitata et al** [83] proposed a visual control design that allows a mini helicopter to hover under local and temporal occlusions. Two fixed and upward-looking imaging devices observe four black balls fixated to rods attached to the

lower side of the helicopter. They have said that the designed structure can hold the helicopter in a resolute hover. **Iwatani et al** [84] presented a visual servo control system using multi-camera for unmanned micro aerial vehicles. The cameras are placed on the floor, and they are connected through a network. They claimed that the controller is durable against to occlusion, and the helicopters can move easily and freely in field of view of the camera. **Weber and Kühnlenz** [85] have utilized triangulation of images obtained by multi-cameras pointing in different directions to control a robot with position based visual servoing (PBVS). They have said that the triangulation is implemented by an iterative linear method which provides high accuracy and real-time operating. **Kermorgant and Chaumette** [86] has offered a basic sensor fusion design for multi-sensor robot positioning. To realize an image-based visual servoing task, two cameras are used in eye-in-device and eye-out-device configuration. It is claimed that this configuration enables a comprehensive comparison of the suggested fusion design of sensor data. **Elsheikh et al** [87] has recommended an application and practical results of dynamic path planning and robot navigation by using visual servoing for a mobile robot in indoor environment. It is said that short locomotion distances for the robot are anticipated, energy consumption is balanced and consequently increase the overall traveling time. **Aliakbarpour et al** [88] have presented a number of contributions to a visual-based mobile robot control by utilizing a general camera model. They said that by utilizing a basic radial model, the suggested visual servoing approach can be employed for a large type of general cameras, both central and non-central. **Ahlin et al** [89] have proposed a leaf grasping system using a robotic manipulator in an unstructured environment by using deep learning and visual servoing. They said that Monoscopic Depth Analysis (MDA) enables for a random number of features in unknown geometric characteristics. **Alepuz et al** [90] have exhibited an visual-based controller to fulfill a robot manipulator guidance. The eye-in-device camera configuration is used for the manipulator and it is placed to a base satellite. The base is entirely independent and floating in space without attitude control. They said that by considering kinematics and dynamics, controller allows the robot to accomplish a input position from an initial one and implement the tracking of a desired trajectory.

Previous studies are generally implemented with internal imaging devices. We mainly focus hybrid utilization of visual features acquired from external imaging

devices and controller method. There are commonly known control methods like PID, Fuzzy controls etc. However, there are no specialized controllers for visual-based control systems. Moreover, most of these researches are usually implemented with only experimental simulations. In addition, all these studies are generally implemented without considering camera scalability in the configuration space. There are no detailed studies about eye-out-device camera configuration. They generally focus stereovision systems with double cameras. Multi camera systems are modeled with PBVS (Position based visual servoing) method commonly. Additionally, the number of studies on eye-in-device are more than the number of eye-out-device studies in the literature. In this thesis study, the eye-out-device multi-camera configuration is investigated. The positioning scheme models for kinematics are designed. The visual based control methods are modeled by using the Gaussian and decision tree methods separately with two (graph and triangle) positioning schemes. All the advantages and drawbacks of the visual based control system with multi eye-out-device camera configuration are presented. We use four cameras for proposed system. However, number of the cameras can be increased. Image stitching process is used only once to create whole map of the configuration space. The path plan is made on this map with A-APF. The path plan is divided according to the relevant cameras. The divided path plans are tracked in each camera separately. The mobile robot motion control is provided depending on the divided path plan under the relevant camera. By this way, multi-image processing problem has been overcome for the mobile robot motion.

3. PRELIMINARY DEFINITIONS

The control of a differential drive WMR is generally performed according to the local and global positions of the robot. A large stream of sensor-based information and encoder data are continuously transferred between the robot and control system. The received sensor data are processed to acquire input for the control model. Therefore, such non-visual robot control systems use a number of resources to implement the dynamic control of the robot [6], [91]. On the other hand, visual-based control process needs less internal/external sensor data on robot, when it is compared to the non-visual control systems. Besides, the position and distance calculation tasks can be carried out by a camera(s) as imaging sensors in VBC systems.

Visual-based control for a WMR hosts a couple of important issues which are camera calibration, object detection and tracking, controlling (real time or offline) and imaging device - robot synchronization [40]. Camera calibration is generally required to scale real-world objects to a 2D plane. If the camera lens is a fish-eye, pin-cushion type than a distortion process may be required for good calibration. Object (robot, obstacles and target) detection is needed to calculate position information of these objects. Tracking objects in real time bounds up with robustness, fastness and efficiency of object detection. Therefore, object detection is a major pillar for the tracking process. Controlling of WMR generally depends on encoder and sensor data in classical methods. However, visual based control heavily depends on momentary image information about its surroundings; thus, image processing techniques are required. By using position information extracted from image frames robot motions/behaviors are modeled. Synchronization means coordination between imaging device and robot. After image device takes a frame, this frame has to be processed before robot makes a motion for a while. Because there will be a time gap between parameter calculation and robot motion.

Except from defined issues; the main problem in visual based approaches with a fixed head camera configuration is that the mobile robot can go out from viewing area of the camera because of an obstacle, etc. Even if it disappears from viewing area, an additional estimation-based method (Kalman etc.) can be utilized to detect the position of robot. A fixed head camera configured visual control system minimizes the errors, because the position of robot is continuously tracked and updated according to obtained information from the taken images. The visual-based

control can ensure significantly stable and accurate results in moving and positioning tasks of the robot control. The specialized control methods for VBC environments are needed. Since, there are no detailed studies for eye-out-device camera configuration modules like positioning scheme, experiment environments etc.

The obstacle avoidance should be considered carefully to create an efficient path plan. The extracted path trajectory has to ensure reaching to the target position without any collision and friction to the obstacles. In other words, there have to be a safe path. The second issue is that the WMR shouldn't fall into a trap in any position at the configuration space. The WMR can exhibit faulty motion behaviors like redundant spinning, oscillation, no motion etc. The third issue is path cost which is a significant issue in terms of energy and time efficiency. If there is a suitable path providing minimum cost, then it should be selected. A path plan is extracted by taking cognizance of these three issues together. Therefore, a safe path with minimal cost and without traps will handle these problematic issues.

Two types of problems may cause to extracting the path plan inadequately in potential field method [92]. Firstly, local minimum problem; when all the attractive and repulsive forces are balanced, the robot falls into a trap and generally motionless. The robot simply doesn't make any progress. Second problem is known as unstable oscillation stemming from high velocity, narrow passages, sudden changes and etc. problems. Because of this problem, the path trajectory is generated with too much swinging. The most basic solutions to these problems are that defining a minimum attractive force on any point in configuration space and creating rotational force fields around the obstacles so that these forces guide moving object. In this study, these problems have been eliminated with adaptive design.

In this study, the mobile robot only fulfills given commands to adjust velocity of wheels. Because the whole control processes are implemented on an external computer system. Therefore, the internal processing device may not be required in mobile robot. Besides, such hardware are generally increases the cost of the system and consume high energy. In this case, a simple command interpreter and a wireless communication infrastructure like Bluetooth or Wi-Fi is enough for eye-out-device configured VBC. Ultimately, the developed methods ensure remarkable performance in aspect of the cost and energy efficiency.

4. MATERIAL AND METHOD

The presented control infrastructure includes several components. Physical environment for experiments, hardware (robot, image processing and computation unit and cameras) and controller software are the primary components. Image stitching, object detection and tracking, parameter acquisition and calculation, path planning and control functions are the main modules of controller software component. Acquired images from the multi-cameras are stitched and environment map is created. The method initially detects and tracks robot control markers and target on this map. A path plan is extracted by considering obstacle positions. Then, the input variables are computed by utilizing positioning scheme model. The velocity parameters are calculated with the developed Gaussian/Decision tree-based control functions. The obtained parameters are transferred to the mobile robot to trigger motion. The controller navigates the robot until final target position is reached.

This thesis study has been implemented in three stages. In first stage; a go-to-goal behavior controller is designed by using Gaussian and Decision Tree methods for an obstacle free configuration space. Two different positioning models have also been developed for the designed controllers. These models provide accurate, fast and smooth controlling process. Each controller-model combination has tested in a real environment with a differential drive mobile robot (or WMR). In second stage; obstacles have been added to the configuration space. Therefore, a path planning strategy is required as well. Artificial potential field (APF) method is modified to design an adaptive path planning method working with dynamic parameters. The adaptive designs have provided suitable parameters to path plan task. The Gaussian or Decision tree-based controllers have been fused with adaptive-APF method to create a novel real-time motion navigation. This new design has been tested with positioning models in an obstacle hosted configuration space. In third and last stage; multi-camera environment has been prepared and mid-controllers (path dividing, path distribution, controller modifications etc.) for this configuration space have been modified/developed. Multi-camera model is used to provide a scalable configuration space. In this way, working space can be expanded. All previous controllers, models and path planners have been tested in multi-camera configuration space as well. The stages are illustrated in Fig. 1.

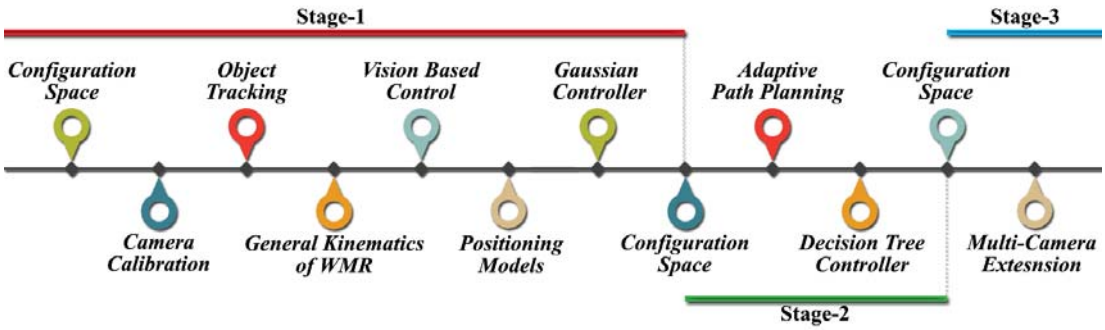

Fig. 1. Stages for the thesis study

4.1. STAGE-1: Go-to-goal Controller

In this stage, a Gaussian based visual controller designed and two positioning models developed for WMR. The controller method has been employed to perform go-to-goal behavior for a differential drive mobile robot. The configuration space is obstacle free and there is one target. This stage includes operating environment, camera calibration, object tracking, general kinematics of WMR and controller method. All these mentioned structures are common for three stages. It can be said that the main base of this thesis study has been formed in this section. The utilized controllers and models are specifically emphasized with details in the stages.

4.1.1. Operating environment of Stage-1

Configuration space consists of four main components; a mobile robot with differential drive feature (or WMR), a target, obstacle free configuration space and a camera with fixed observing (bird's eye view) configuration. The visual-based control task is implemented on a floor with plain color covering under variable light intensity values. The vertically hanged fixed camera is placed 180 cm above from the floor. The camera has 3.2 MP resolution and its lens has 0.3 mm focal length. SVGA (Super VGA: 800x600; 4:3) resolution is used. MATLAB R2016a is used for both image processing and robot controlling process. Communication with the camera is performed through USB 3.0 interface. The mobile robot and computer system communication is carried out over Bluetooth 2.0 standards. Experiments have been implemented on Intel i3-3217U 1.80 GHz CPU with 6GB DDR3 1600 MHz Memory and 5400 Rpm HDD. Operating environment (configuration space) is demonstrated in Fig. 2.

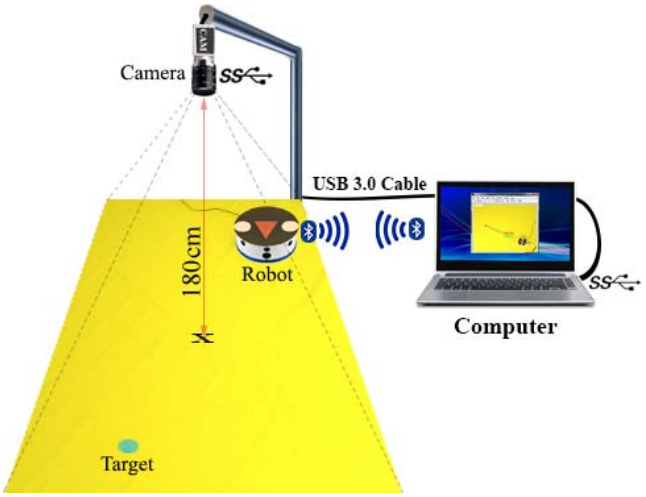

Fig. 2. Operating environment for proposed control system

4.1.2. Camera Adjustment and Image Distortion

A general perspective of projection model for a camera is shown in Fig. 3. Velocity screw of the image frame is defined in (4.1) and (4.2). The w is rotational velocity and $v(O)$ is translational velocity in this equation [93]. If it is assumed that the focal length of the imaging device is equal to one – '1' , then a point with $\vec{x} = (x\ y\ z)^T$ coordinates is projected to a plane on the input image as a point with $\vec{X} = (X\ Y\ 1)^T$ by utilizing (**Hata! Başvuru kaynağı bulunamadı.**).

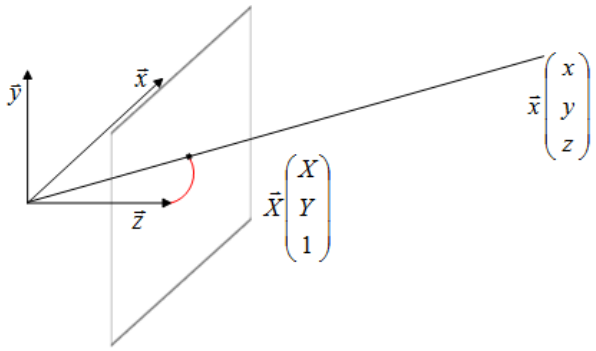

Fig. 3. General perspective of projection model for a camera

$$F_s(O, \vec{x}, \vec{y}, \vec{z}) \rightarrow T = (v(O), w) \tag{4.1}$$

$$\vec{X} = \frac{1}{z}\vec{x} \tag{4.2}$$

Camera calibration is required to scale and fit image frames to the 2D plane. A fish eye lens is used. It transfers images to the imaging device sensor as barrel type. Therefore, a distortion process is required to extract visual information from images properly. Image planes are demonstrated with I-Barrel type, II-Pincushion type and

III-Distorted in Fig. 4. In barrel type images; the magnification of image declines with distance from the optical axis that stem from the focal length of camera lens [93]. Model for radial image distortion is expressed with (4.3), (4.4), (4.5) and (4.6) used to project images to two dimensional plane. The distorted Cartesian coordinates x_d and y_d, distortion parameter is κ, actual image coordinates are x_a and y_a, f_x and f_y are focal length (mm), image center (principal point) are c_x and c_y.

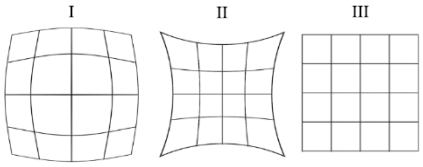

Fig. 4. Image planes (I. Barrel, II. Pin-Cushion, III. Distorted)

$$x_d = x(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6) \tag{4.3}$$

$$y_d = y(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6) \tag{4.4}$$

$$r^2 = x^2 + y^2 \tag{4.5}$$

$$x_a = f_x x_d + c_x \, , y_a = f_y y_d + c_y \tag{4.6}$$

4.1.3. Object Tracking

Performing a good visual based control depends on performing a good object detection and tracking. Because quality of this detection and tracking are decisive factors for success of visual based control process. After detecting the obstacles and target point in first frame, it is not needed to detect these components anymore for a static environment (there are no dynamic objects). Therefore, only the robot is detected and tracked in following image frames. Quantization and color thresholding is used to perform detection process of labeled target, obstacles and robot. At first step, an image frame is acquired with the fixed camera. Secondly, color space of this image is transformed from the RGB space to the HSV space. Then a color mask filter is executed to segment the related objects. Maximum and minimum range of the Hue, Saturation, and Value channels are used in this mask function. HSV separates the luminosity (or the image intensity), from the chrominance (or the color information). This is very efficient in remarkable number of applications [94]. Therefore, if it is wanted to separate the color components from the intensity for various reasons, then HSV is a good choice against these issues in computer vision.

These reasons are robustness to illumination (changes in light level), get rid of shadows etc. In next step, colors of image frame are quantized by decreasing color depth 8 bit to 256. Then objects are detected by using color thresholding process. Each acquired frame is exposed to the color thresholding process. After detecting the position information of each object, the new velocity parameters for the WMR are calculated.

The thresholding process is implemented with several steps: Each cell of the whole image matrix is controlled. The cells having threshold values in these three-channel transmuted to '1' and the remaining cells transmuted to '0'. It simply filters the color values by controlling the predefined threshold ranges. This procedure known as "binary image acquisition" [95]. Equation (4.7) is utilized to identify range of color channels defined in mask function. The $h(p_i)$ corresponds to the color channel histograms in this equation. p_k expresses the kth channel level and M corresponds to the total level of the channel. To purify image from noisy parts which have pixel value below the defined threshold, an elimination method converts these noises to '0'. Coordinate values of detected object centroids are computed. Then, coordinate values are recorded to storage unit and marked on the related image frame. The detection steps are demonstrated in Fig. 5. The image processing and controlling scheme is given in Fig. 6.

$$f(p_k) = \frac{1}{M}\sum_{i=1}^{M} h(p_i)K_\sigma(p_k - p_i), \qquad p\epsilon\{H,S,V\} \tag{4.7}$$

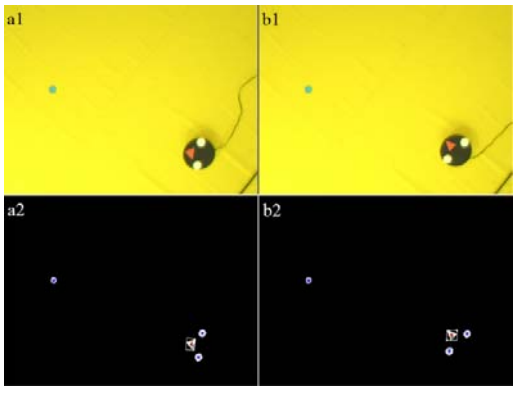

Fig. 5. (a1-b1) Real time image frames from different experiments, (a2-b2) Centroid detection of object components

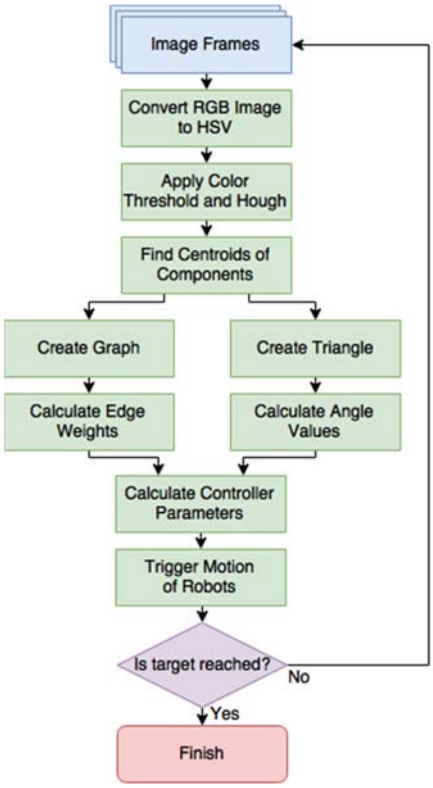

Fig. 6. Image processing and controlling diagram of visual-based control

Color quantization is a kind of clustering method that groups similar colors within a threshold value [96]. It is used to dissociate color values and sharpen similarities between color information of objects and other components in the image. It simply facilitates the tasks of color thresholding masks before detection. Because of any adaptive color-based thresholding method has not been used, there can be faulty detections for objects when illumination changes remarkably. In other words, light changes affect the luminosity value of each image frame. Therefore, quantization process has been used to decrease the negative effects of this sudden light changes. By assuming that the quantization levels of Q as $q_j, j = 1,\dots,n$, the quantization cells can be computed. Suppose that in the Voronoi diagram for Q, the C_i are the cells, then following the term (4.8) equation is formed to minimize expected error. In this equation, the distortion parameter is d and the quantizer is q. By assuming having the C_j quantization cells, the optimal value for q_j is defined by using (4.9).

$$E(d,q) = \int_c d\big(x,q(x)\big)d\mu = \sum \int_{c_j} d\big(x,q(x)\big)d\mu \geq \sum \int_{c_j} d\big(x,q_j\big)d\mu \qquad (4.8)$$

$$q_j = \arg\min_{c\in C, x\in C_j} E(d(x,c)) = \arg\min_{c\in C} \int_{C_j} d(x,c)d\mu \qquad (4.9)$$

The mobile robot tracking is a key issue to collect information about the global position of the robot, accurately. Since this information is utilized to calculate wheel velocities input parameters. Instead of processing entire image frame as an input to segmentation method; local frames (LF) covering the robot has been formed to enhance the performance of segmentation. This structure is created according to the dimensions of robot. The local frame creation process is demonstrated in Fig. 7. The coordinates of local frame are calculated between certain numbers of image frames according to a pre-defined threshold parameter. The (4.10) and (4.11) equations are employed to implement this process. In these equations; the n represents the frame numbers acquired. The global coordinates of the robot are x_l and y_l (P1 point). The global coordinates of local frame are LF_x and LF_y and equal to x_l and y_l initially. So, the robot center node is also utilized to ensconce local frame around the robot in specific image frames. The new global coordinates for the robot are x_{l_n} and y_{l_n}. LF coordinates are only updated in every 15 image frames. The LF is created by starting from a position (P2 point) calculated with the pre-defined distances. The LF covers the area starting from $x_l - 90$ and $y_l - 90$ (red point) to a constant equal length for two axes.

$$LF_x = \begin{cases} x_{l_n}, & if\ n\%15 = 0\ and\ n > 0 \\ x_l, & if\ n\%15 \neq 0\ and\ n > 0 \end{cases} \tag{4.10}$$

$$LF_y = \begin{cases} y_{l_n}, & if\ n\%15 = 0\ and\ n > 0 \\ y_l, & if\ n\%15 \neq 0\ and\ n > 0 \end{cases} \tag{4.11}$$

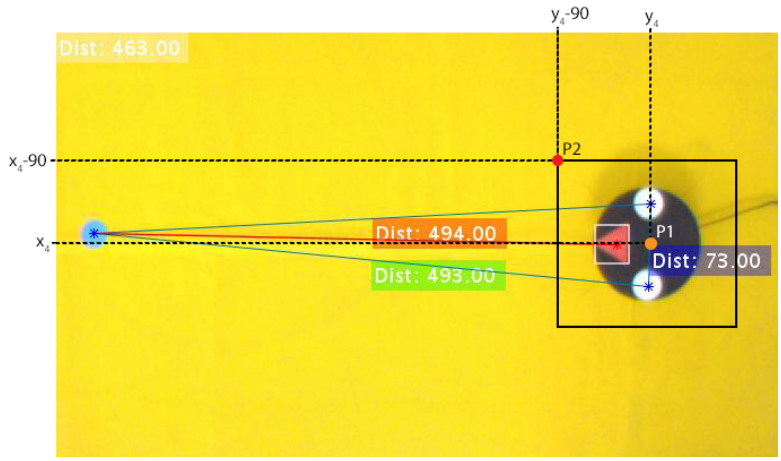

Fig. 7. Local frame demonstration on a real image frame

4.1.4. General Kinematics of WMR

Before touching the control scheme in thesis study, it is suitable to give an idea about general control scheme of a WMR. The general kinematics are simply emphasized in [97]. Assume that the position of a WMR is selected as (x_d, y_d) according to the center of the target position (x_r, y_r). The L distance and φ angle can be set to be simply constant parameters. Then the position parameters of the robot-target can be explicated with following equations (4.12) and (4.13).

$$x_d = x_r + L\cos(\theta_r + \varphi) \tag{4.12}$$

$$y_d = y_r + L\sin(\theta_r + \varphi) \tag{4.13}$$

The v_c (v_x, v_y) velocity and ω_c (ω_l, ω_r) angular velocity are directly bound up with to the velocities of the right and left wheels of the mobile robot. Angular velocity values $\omega_l(t)$ and $\omega_r(t)$ are calculated by using equation (4.14). In this equation r represents the wheel radius and l represents the distance between the two wheels (wheelbase).

$$\begin{bmatrix} v_x(t) \\ v_y(t) \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} r/2 & r/2 \\ 0 & 0 \\ -r/l & r/l \end{bmatrix} \begin{bmatrix} \omega_l(t) \\ \omega_r(t) \end{bmatrix} \tag{4.14}$$

To make compatible the kinematic models to the real-world pattern following equations (4.15) and (4.16) are used. These equations characterize velocity and angular velocity parameters for the mobile robot. In this equation R parameter is instantaneous curvature radius of the robot trajectory relative to the mid-point axis.

$$v(t) = \omega(t)R = \frac{1}{2}(v_r(t) + v_l(t)) \tag{4.15}$$

$$\omega(t) = \frac{v_r(t) - v_l(t)}{l} \tag{4.16}$$

A general expression can be put forth for velocity (v) and angular velocity (ω) parameters. By this way, we can simply generate following final equations (4.17) and (4.18) for a two-wheeled differential drive mobile robot.

$$v = \frac{R}{2}(v_r + v_l) \rightarrow \frac{2v}{R} = v_r + v_l \tag{4.17}$$

$$\omega = \frac{R}{L}(v_r - v_l) \rightarrow \frac{\omega L}{R} = v_r - v_l \tag{4.18}$$

The general kinematics of the differential drive mobile robot can be characterized with the following equation (4.19). The v_c is linear forward velocity and ω_c is angular velocity for the WMR in this equation.

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta}_c \end{bmatrix} = \begin{bmatrix} v_c \cos\theta_c \\ v_c \sin\theta_c \\ \omega_c \end{bmatrix} \tag{4.19}$$

Similarly, the kinematics for the target reference can be modeled with the following equation (4.20).

$$\begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} v_r \cos\theta_r \\ v_r \sin\theta_r \\ \omega_r \end{bmatrix} \tag{4.20}$$

By fusing previous statements, we can extract control variables with the following equation (4.21). Then the control parameters are defined with the equation (4.22) statement.

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos\theta(t) & 0 \\ \sin\theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} \tag{4.21}$$

$$\dot{q}(t) = S(q)\xi(t) \tag{4.22}$$

4.1.5. Vision Based Control

Visual based control is one of the popular research area in robotics. It is also known as visual servoing. It simply concerns about controlling a robot by using visual information around its environment. In other words; visual based control methods aim to manage a dynamic system by using visual features obtained from images provided by one or multiple cameras [9], [98], [99]. Except from conventional sensors visual based information is a necessity for robotics. For instance, real-time robotic systems, human-robot interaction, autonomous vehicles, indoor and outdoor robotics etc. are almost all utilize the visual information to make a decision. It is the major part of human/animal inspired control theory. Since most of the livings utilize their visual sensory organs predominantly besides their other sensory organs. Of course, simple

tasks, repetitive works, trajectory tracking etc. simply can be done by using non-visual sensors (encoders, odometry, altitude, gyro, accelerometer, velocity sensors and so on) alone. But when a recognition process is an inevitable situation then visual based control task becomes imperative partially or totally.

In Fig. 8, the main steps for image processing until the controlling process of the mobile robot are shown as layered structures. It simply shows; image acquisition (a), object detection (b), the graph/triangle construction (c), tracking of local frame (d), parameter calculation for dynamics (e), and motion of mobile robot (f).

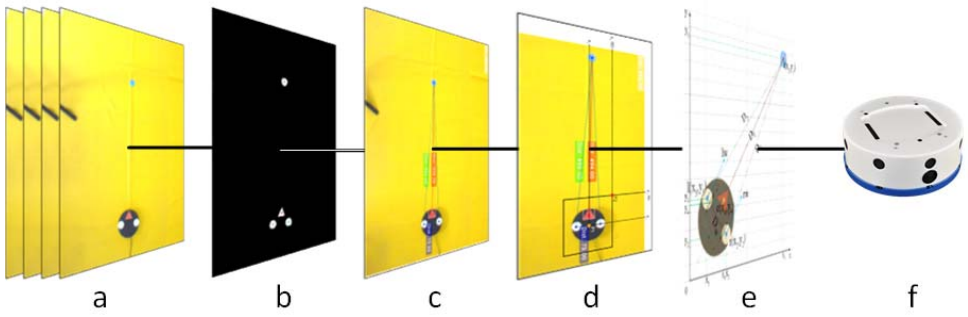

Fig. 8. Main image processing steps in control process

4.1.6. Positioning Models of Kinematics for Proposed Models

Detected objects from the segmentation process are expressed as base components in the binary image. Each centroid of these base components represents control points for the mobile robot control models. These control points are connected to each other with line parts. The control points and lines are utilized as nodes and edges for graph-based positioning model as the first approach. Distance values of the lines have been computed and appointed as edge weights. These values are input to the control process for graph-based positioning model. In second positioning model, triangle structure is formed by utilizing these control points and lines. Interior angle values are computed between line intersections. These values are used as input to the control process for the triangle-based model. Details of the positioning models are given separately in following section.

A. Weighted graph-based model

Positioning Scheme for Kinematics

In Fig. 9 (a), graph-based positioning model for the control method is shown. In this figure, the node ‘t’ corresponds to the target, ‘r’ and ‘l’ nodes represent right and

left circular labels (control points of wheels), the 'c' node expresses the centroid of red triangle label. 'D_1', 'D_2', 'D_T' and 'D_W' specify edges between nodes and they have distance values. The distance value of 'D_T' is the length of the shortest path from initial position to the target position. These distances are weights of each edges in the graph. They have been used as inputs to perform control task for mobile robot. Each node has 2D Cartesian coordinates and except for target node 't', coordinates of other nodes change as the motion is performed. The 'lw' and 'rw' represents orientation of the robot wheels. Segmented base components (graph nodes) of a real image frame are demonstrated in Fig. 9 (b). RGB labels correspond to the distance values (weights) of each edge. For example, the distance value between the target (t) and left node (l) on the robot is '771px'. The obtained graph parameters are updated in each image frame with respect to the new values of the node coordinates. The velocity of left and right wheels are successfully set in each updating process in real time.

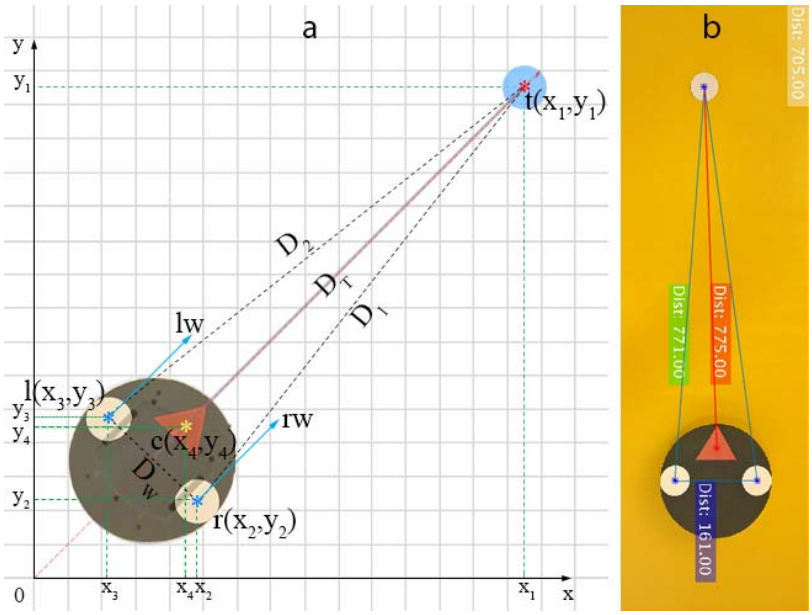

Fig. 9. (a) Distance based positioning scheme, (b) A real image frame

Distance Input Calculation

Euclidian Distance Value (EDV) is computed and assigned as a weight value to each edge between the nodes (t, l, r and c) in the graph by using (4.23). EDV of the D_T edge expresses the distance from robot to the target. EDVs are assigned to D_1, D_2, D_T and D_W in each control loop. The EDV values are utilized to characterize the input parameters of the controller algorithms/functions.

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \tag{4.23}$$

B. Triangle Based Model

Positioning Scheme for Kinematics

The triangle-based positioning model for control method is shown in Fig. 10 (a). The 't' point corresponds to the position of target, 'l' and 'r' locations express the left and right wheel labels, the 'c' point of the triangle component represents the centroid of directional label in this figure. 'A_L – Angle of Left Wheel', 'A_R – Angle of Right Wheel' and 'A_T – Angle of Target' specify the angles between edges. These angle values are simply interior angles of a polygonal shape (triangle). They are primary input variables to implement mobile robot control process. Each corner point of triangle has coordinates in 2D space and these coordinates and angle's values change while robot moves. The robot direction is modeled according to the changing A_L and A_R difference in each control cycle. An example real-world image frame for detected objects and triangle positioning is shown in Fig. 10 (b). The degree values of each angles have been shown with the color labels. For instance, degree at the left label is '85.33°'. The angle values at target and wheel labels are calculated and updated in each control cycle. The velocities of wheels are accurately calculated by using these angle values in control process.

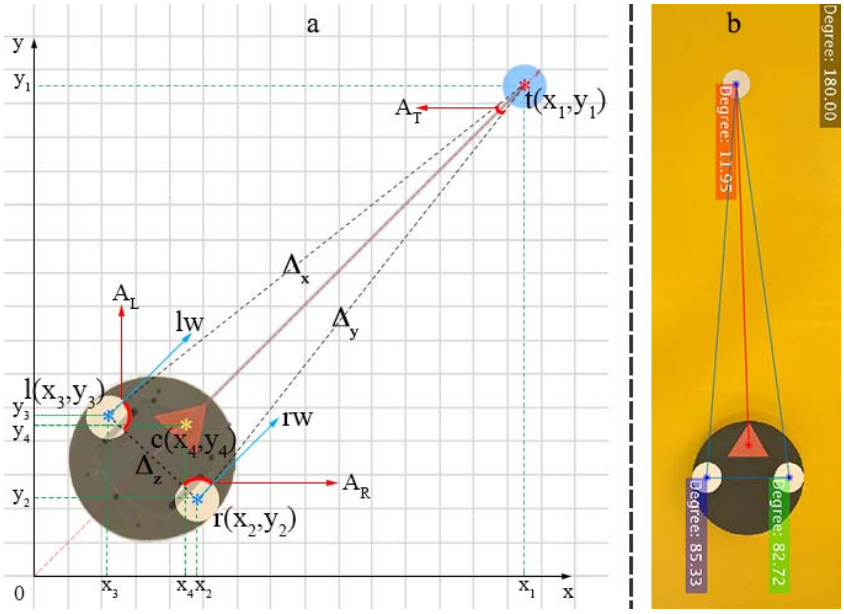

Fig. 10. (a) Angle based positioning model scheme, (b) A real image frame

Angle Input Calculation

Each intersectional edge has an angle value in triangle structure. These angle parameters are determined by utilizing (4.24) and (4.25). The Δ_x, Δ_y and Δ_z are distance vectors between 't', 'l' and 'r' coordinates of discovered objects. The ang_r is the radian value of related angle. The A_{360} corresponds to the degree equivalent of the radian value. Velocities for both wheels are calculated by using these angle values for triangle-based approach.

$$ang_r = acos\left(\frac{\Delta_x^2 - \Delta_y^2 - \Delta_z^2}{-2 * \Delta_y * \Delta_z}\right) \tag{4.24}$$

$$A_{360} = ang_r * \frac{180}{pi} \tag{4.25}$$

C. Unit Transformations for Velocity of Wheel

The pulse values are sent to the robot to control velocity of wheels. The received pulse values are converted to *mm/s* unit with (4.26). The V_w represents velocity of the left-right robot wheels in mm/s. The v_p represents pulse value sent to the wheels. The t_r corresponds to the refresh time (a constant value). The $\emptyset_w$ express the diameter of the wheel. The Nb_p represents the cycle resolution.

$$V_W\left[\frac{mm}{s}\right] = \frac{v_p}{t_r} * \frac{\emptyset_w * \pi}{Nb_p} \tag{4.26}$$

4.1.7. Gaussian Control Model Kinematics

Gaussian function is a smoothing method which is commonly used in a great number of studies. The general Gaussian function with single dimension (f_G) is defined in (4.27). The parameter x is the difference of input parameters ($D_1 - D_2$ or $A_L - A_R$). The π value (~3.14) is a constant parameter in f_G. The σ is standard deviation, variance is σ^2 and mean or expectation of the distribution is μ. The value of σ is empirically adjusted to '0.41'. The μ value is set to '0'. Since the optimal x value is '0', it means that both D_1 and D_2 or A_L and A_R are equal or the robot direction is straight toward the target. The controller input x is the absolute difference value between D_1 and D_2 distances for graph approach (4.28). Moreover, this x input is the absolute difference value between A_L and A_R angles for triangle approach (4.29). The

λ and φ parameters are used as smoothing factors. If x value approximates to '0', then f_G approximates to '1' and if x value approximates to '∞', then f_G approximates to '0'. Therefore, S_C – "Speed Coefficient" is computed by multiplying S_{max} – "maximum speed of wheel" with the subtraction of the f_G value of related wheel from '1' by using (4.30). Equations (4.31), (4.32) and (4.33) are used to find S_L - 'Speed of Left Wheel' and S_R – 'Speed of Right Wheel'. The S_{max} is the pulse value. It can be set to '1200' for utilized mobile robot as maximum tick value. The 'τ' is a constant scaling coefficient. Acquired value of f_G function provides addition to the velocity value of one wheel and subtraction from the velocity value of the other wheel according to sign of x. The f_G is employed to characterize velocities of the robot wheels with a single function. It is used to handle control of a differential drive mobile robot. The f_G is adapted to the configuration space to perform real time go-to-goal control in a static environment.

$$f_G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{4.27}$$

$$x = \left|\frac{D_1 - D_2}{\lambda}\right| \tag{4.28}$$

$$x = \left|\frac{A_L - A_R}{\varphi}\right| \tag{4.29}$$

$$S_C = S_{max} * (1 - f_G) \tag{4.30}$$

$$S_L = \begin{cases} S_{max} * \tau + S_C, & D_1 < D_2 \\ S_{max} * \tau - S_C, & D_1 > D_2 \end{cases} \tag{4.31}$$

$$S_R = \begin{cases} S_{max} * \tau + S_C, & D_1 > D_2 \\ S_{max} * \tau - S_C, & D_1 < D_2 \end{cases} \tag{4.32}$$

$$S_{L,R} = S_{max} * \tau + S_C, \quad D_1 \cong D_2 \tag{4.33}$$

The advantage is that a single f_G model is adequate to set velocity parameters for mobile robot wheels in every control process cycle. It ensures a fast and efficient processing performance in real time. The output of this f_G model is added to the velocity of the behindhand wheel and subtracted from the velocity of the leading wheel of the mobile robot. This means that the behind wheel according to target position, takes positive f_G value for its velocity calculation. In this way, velocity

values of wheels are affected from position values. Fig. 11 shows our f_G curves. The red line indicates the x input velocity when the robot is in motion state and the blue line indicates the stopping process with $f_G = 1$.

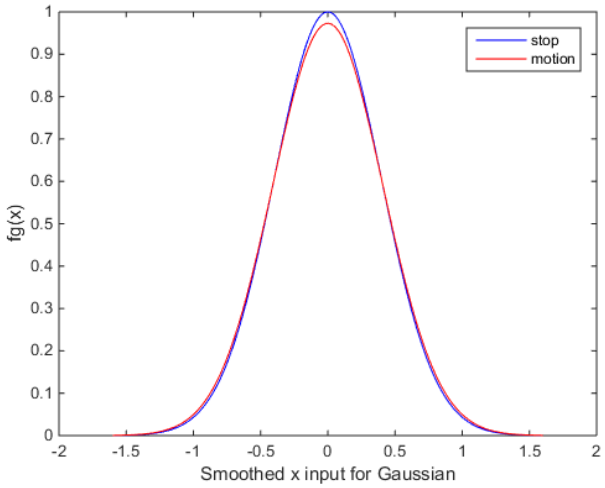

Fig. 11. Gaussian curve graphics for designed control

A. Calculation of Path Cost Difference

The path cost difference is generated between the linear D_T path and the real path created by the robot motions. The path cost difference is simply calculated by using (4.34). This value shows how much difference occurs between these paths. The D_T represents the linear path distance and the D_A corresponds to the actual path distance. The sampled coordinates on the formed path are the $x_1y_1 \dots x_ny_n$. The actual point is q_x and the next sampled point is p_x.

$$\frac{D_T}{D_A} = \frac{\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}}{\sum_{x_1y_1}^{x_ny_n}\sqrt{\sum_{j=1}^{m}(q_j - p_j)^2}} \tag{4.34}$$

B. Procedures of Stopping Process for Kinematics

The stopping task is performed when a predefined threshold is reached. This threshold parameter is the distance value D_W between r and l coordinates on the robot label in *graph approach*. When the distance between robot and target (D_T) equal to or below this threshold by reducing gradually, '0' signal pulse is transmitted to the robot with $S_{L,R}$ calculation by changing parameter to start the stopping process of the robot. The D_T is controlled in each control loop. This threshold parameter is controlled according to the target angle value A_T at the target position in *triangle approach.*

When the A_T value is equal to or exceeds the '60^o' by increasing gradually, '0' signal pulse is transmitted to the robot with $S_{L,R}$ calculation by changing parameter to start the stopping process of the robot. Similarly, the A_T is controlled in each control loop.

In experiments, the mobile robot has stopped according to the target and robot positions. The f_G parameter value σ is reassigned in the end of the execution time passed to complete given task. This reassigning process of σ parameter is performed to stop the robot in a certain stage when previously mentioned threshold value(s) is reached. The robot motion is stopped with '$f_G = 1$' value by changing value of x to '0' and value of σ to '$1/\sqrt{2\pi}$'. This means that $S_{L,R}$ takes the '0' value and so the stopping process of the robot is triggered with (4.35) for the graph and (4.36) for the triangle approaches.

$$S_{L,R} = S_{max} * S_C, \qquad D_T \leq D_W - 30 \tag{4.35}$$

$$S_{L,R} = S_{max} * S_C, \qquad A_T \geq 60 \tag{4.36}$$

If this threshold condition is not met for the utilized control approach, then the calculation of velocity is proceeded with the previously defined (4.31), (4.32) and (4.33) equations. The computed velocity values are converted to the tick/pulse type with the parameter transformations and then transmitted to the robot.

C. Control Scheme for Visual Control System

The entire control stages for controller models is demonstrated in Fig. 12. An image frame captured from the fixed head camera in real-time. Object detection tasks have been executed in the captured image frame. A weighted graph or triangle structure is formed according to the utilized model approach by using identified objects. The weight values (distances) of edges for graph model and the degree values (angles) of edge intersections for triangle model are calculated. The Gaussian function is operated by giving 'x' difference value (D_1-D_2 or A_L-A_R) as input variable. The output of the Gaussian is added/subtracted from wheel velocity values according to this 'x' value sign ($-$ or $+$). The $(+,-)$ signs means that the velocity of the left wheel increased and the right wheel is decreased. Similarly, $(-,+)$ signifies the vice versa. Besides, $(\sim,\sim)$ corresponds that the wheels are affected by same rate. The threshold of stopping process is controlled, if the threshold is satisfied, then the control procedure stops. Otherwise, control processes continue by taking next image frame

from the initial phase. These entire tasks are performed by utilizing only the visual features without any information from the sensors.

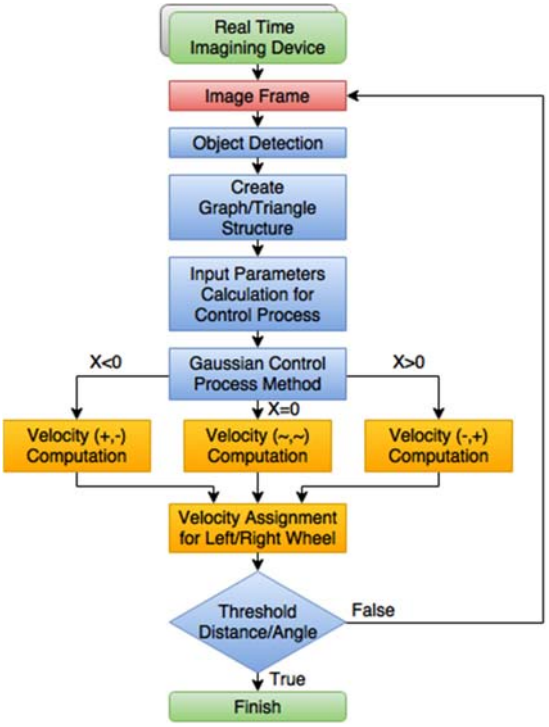

Fig. 12. General working phases for designed control method

4.2. STAGE-2: Path Planning

4.2.1. Operating environment of Stage-2

As distinct from Stage-1, the operating environment includes obstacles for Stage-2. Therefore, a path planning method have to be used to extract a proper trajectory for WMR. The camera position and system hardware are all have same parameters as in previous stage. The obstacles are placed with the several different configurations to see performance of the control infrastructure in simple and extreme conditions. The operating environment for Stage-2 is demonstrated in Fig. 13

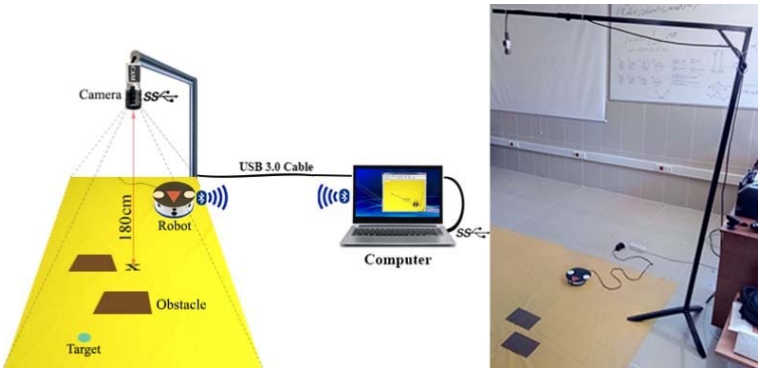

Fig. 13. Operating environment (Left – Representative, Right – Real)

Fig. 14 shows the object detection task for this stage. There are three kind of object that have to be detected. These objects are used to perform tracking process in mobile robot control process. To detect obstacles, a suitable threshold mask is used. Because of static nature of the configuration space, obstacles are detected for once at the start of the control process. After that only the mobile robot is tracked to detect its new position. The target and obstacles are static, so it is not necessary to track them. Their positions have been simply stored for further usages.

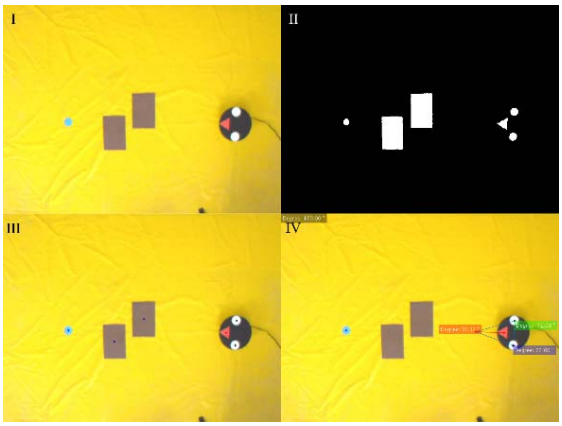

Fig. 14. Object Detection; I. Acquired image, II. Thresholded image, III. Detected objects, IV. Calculated angles

Local frame method is utilized exactly same as in previous stage. Local frame coordinates are updated when a threshold limit for processed image frames is reached. Fig. 15 demonstrates local frame detection in configuration space.

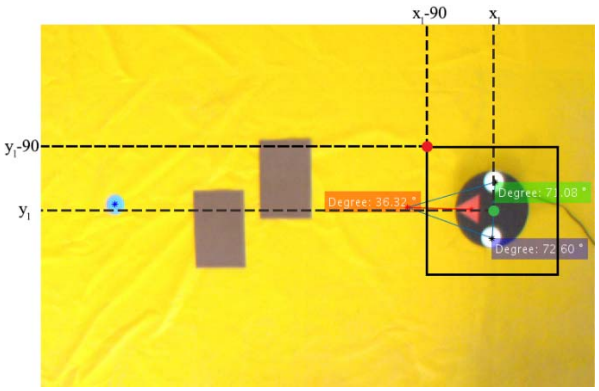

Fig. 15. Local frame demonstration on a real image frame

4.2.2. Fundamentals of Potential Fields

Potential field is a term used to define vectors array representing a certain space. Typically, a vector consists of magnitude (m) and direction (d) parameters, Fig. 16. A vector represents a force in potential fields. Length of the vector corresponds to the

magnitude and angle of the vector corresponds to the direction. All these vectors in potential fields can be resembled offline local compasses giving the needed direction with a distance weight at different points. Therefore, the Artificial Potential Field (APF) can be utilized to model go-to-goal and avoiding obstacles behaviors together. Two kinds of force are used to steer and advance the object (robot) in APF method. These forces are attractive field and repulsive field forces. Potential field structures are illustrated in Fig. 17 for an electrical potential field; (a) represents attractive field between opposite poles and (b) represents repulsive field between same poles.

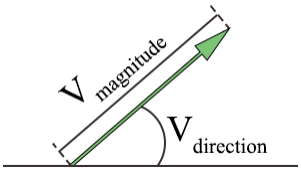

Fig. 16. Vector parameters

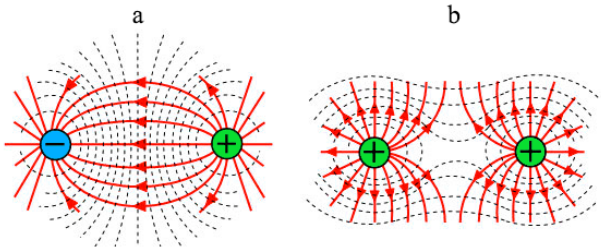

Fig. 17. Potential field structure (Electrical)

Attractive Potential Field (P_{att}): It has attractive forces used to attract/pull the object to the target position according to the given configuration space. Magnitude of this force generally proceeds with same value until the moving robot reaches to the target position. According to design this attractive force can be increased or decreased under certain circumstances. It can be linearly increased or decreased or dynamically changed according to configuration space parameters. Repulsive Potential Field (P_{rep}): It has forces pushing object at a certain rate to overcome obstacles and unseen locations. Magnitude of this force can generally show variability according to size of the obstacles and distance values between object and obstacles. In other words, this force demonstrates a dynamic variability until robot reaches to the target position. Gradient vector structures are utilized while these forces are specified as magnitude and direction. These forces are demonstrated in Fig. 18 for a representative configuration space. F_{att} is the attractive force, F_{rep} is the repulsive force and F_{total} is the resultant sum of these two forces.

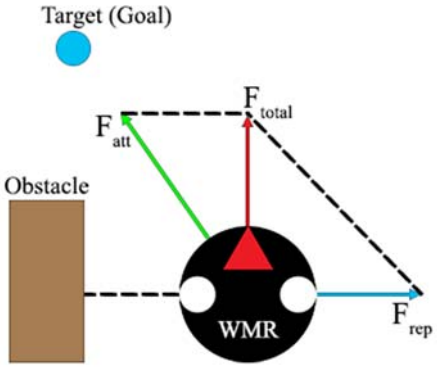

Fig. 18. Potential field forces

In some cases, a vector field is gradient of a $P(x, y)$ potential function expressed as following (4.37);

$$(\Delta x, \Delta y) = \nabla P(x, y) = \left(\frac{\partial P}{\partial x}, \frac{\partial P}{\partial y}\right) \tag{4.37}$$

If the potential area is defined mathematically, $q = (x, y)^T$ to be the robot coordinates, then it is basically computed by utilizing (4.38). $P_{total}(q)$ is total potential field in configuration space. $P_{att}(q)$ is attractive filed and $P_{rep}(q)$ is the repulsive field in this equation.

$$P_{total}(q) = P_{att}(q) + P_{rep}(q) \tag{4.38}$$

Attractive force is the negative gradient of attraction field and repulsive force is the negative gradient of repulsion field. $F(q)$ is to be gradient of artificial force vector field, then it is found by using (4.39).

$$F(q) = -\nabla P_{att}(q) - \nabla P_{rep}(q) = F_{att}(q) + F_{rep}(q) \tag{4.39}$$

In this equation, ∇P expresses P gradient vector. $F_{att}(q)$ is the artificial attractive force and $F_{rep}(q)$ is the artificial repulsive force. Robot position is shown with $q(x, y)$. Thus, the following equality (4.40) is found.

$$F_{att}(q) = -\nabla P_{att}(q) \text{ and } F_{att}(q) = -\nabla P_{rep}(q) \tag{4.40}$$

Attractive field between target and robot is built to attract robot to the target position. Attractive potential field created by target is found with (4.41).

$$P_{att}(q) = \begin{cases} \frac{1}{2}\zeta d^2(q, q_h), d(q, q_h) \leq d *_h, \\ d *_h \zeta d(q, q_h) - \frac{1}{2}\zeta d *_h{}^2, d(q, q_h) > d *_h, \end{cases} \quad (4.41)$$

$d(q, q_h)$ is current distance value of robot to the q_h target. ζ is attraction gain and $d *_h$ threshold value being next value of conical form of second-order function.

$q_g = (x_g, y_g)^T$ to be location of target vector. $\rho_{goal}(q) = \|q - q_g\|$ is the Euclidean distance from the position of robot to the position of target. Attractive force on the robot is computed as negative gradient of attractive potential field and gets the form (4.42) given below.

$$F_{att}(q) = -\nabla P_{att}(q) = -\frac{1}{2}\zeta \nabla \rho_{goal}^2(q) = -\zeta(q - q_g) \quad (4.42)$$

$F_{att}(q)$ is a vector which head toward q_g point (target/goal). It has linearly related magnitude to distance from q to q_g. The $F_{att}(q)$ components are negative directional derivatives of the attractive field throughout x and y directions in 2D coordinate space. Therefore, when influencing of attractive potential field starts, components can be expressed as (4.43) and (4.44);

$$F_{att-x}(q) = -\zeta(x - x_g) \quad (4.43)$$

$$F_{att-y}(q) = -\zeta(y - y_g) \quad (4.44)$$

In these equations, F_{att-x} is attractive force in x direction and F_{att-y} is attractive force in y direction. The mobile robot has to be pushed from the obstacles. The influence on the robot by obstacles is not wanted when robot is far away from these obstacles. Then, repulsive potential field (4.45) can be formed as following.

$$P_{rep}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(q)} - \frac{1}{Q^*}\right)^2, \rho(q) \leq Q^* \\ 0, \rho(q) > Q^* \end{cases} \quad (4.45)$$

In this equation, η is repulsion gain, Q^* is the distance threshold value that will create a repulsive force on robot for an obstacle. $q_c = (x_c, y_c)$ to be a unique configuration for the nearest obstacle to q. The shortest path between the robot and obstacle is $\rho(q) = \|q - q_c\|$. When distance between obstacle and robot is larger than Q^*; there is no influence for robot. Note that, ζ and η gain parameters and d* and Q^*

threshold parameters are set empirically. Similarly, repulsive force is the gradient of repulsive potential function given as following (4.46) and (4.47).

$$F_{rep}(q) = -\nabla P_{rep}(q) = \begin{cases} 0, & \rho(q) \geq \rho_0 \\ \eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)\left(\frac{1}{\rho^2(q)}\right)\nabla\rho(q), & \rho(q) \leq \rho_0 \end{cases} \quad (4.46)$$

$$F_{rep}(q) = \begin{cases} 0, & \rho(q) \geq \rho_0 \\ \eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)\left(\frac{1}{\rho^2(q)}\right)\left(\frac{\mathrm{q} - \mathrm{q_c}}{\|q - q_c\|}\right), & \rho(q) \leq \rho_0 \end{cases} \quad (4.47)$$

$F_{rep-x}(q)$ and $F_{rep-y}(q)$ are the repulsive forces Cartesian components. When the repulsive field become effective on the robot, the components can be expressed as (4.48) and (4.49).

$$F_{rep-x}(q) = \begin{cases} 0, & \rho(q) \geq \rho_0 \\ \eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)\left(\frac{1}{\rho^2(q)}\right)\left(\frac{\mathrm{x} - \mathrm{x_c}}{\|y - y_c\|}\right), & \rho(q) \leq \rho_0 \end{cases} \quad (4.48)$$

$$F_{rep-y}(q) = \begin{cases} 0, & \rho(q) \geq \rho_0 \\ \eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)\left(\frac{1}{\rho^2(q)}\right)\left(\frac{y - y_c}{\|y - y_c\|}\right), & \rho(q) \leq \rho_0 \end{cases} \quad (4.49)$$

Although there are a great number of obstacles, repulsive potential field in total is sum of repulsive potential fields of all obstacles. The potential field in total can be signified with following equality (4.50);

$$P(q) = P_{att}(q) + \sum_{i=1}^{n} P_{rep}(q) \quad (4.50)$$

In this equation, n corresponds to the number of obstacles. Then, total artificial force can be formed as following (4.51);

$$F(q) = F_{att}(q) + \sum_{i=1}^{n} F_{rep}(q) \quad (4.51)$$

On the other hand, there are several issues which cause that the method not to work properly in potential field. The moving object cannot make any progress because of local minimum or it makes excessive swinging in its movements due to unstable oscillation. In Fig. 19, local minimum (I, II) and unstable oscillation (III) problems are demonstrated. As seen in figure, local minimum problem stems from balanced

repulsive and attractive forces. Because of this balancing situation robot cannot make any progress, so it cannot reach to the desired position. Unstable oscillation problem stems from narrow passages or unsuitable parameters. The robot makes excessive and unnecessary swinging movements, so it reaches to the target very late and wastes energy.

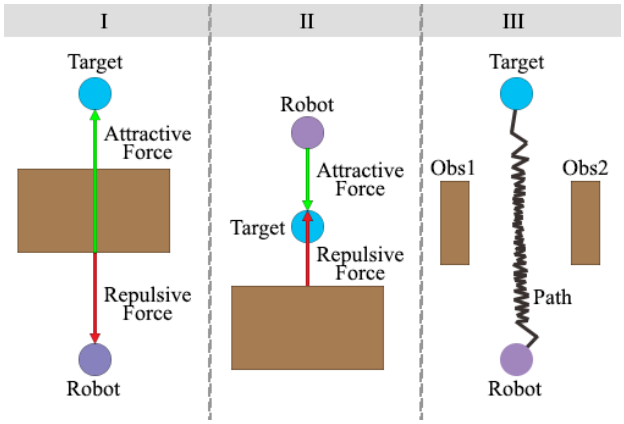

Fig. 19. Main APF problems (Local minima and unstable oscillation)

4.2.3. Adaptive Artificial Potential Field (A-APF)

These mentioned parameters are dynamically changed in proposed adaptive artificial potential field method. Parameters are readjusted depending on magnitude of potential forces from the obstacle, global positions of obstacles and distance to target parameters in each new path coordinate assignment and control iteration step. Objects and environment variables are demonstrated in Fig. 20. Blue arrows represent attractive force and red arrows represents repulsive force. The equation (4.52); $d(q,q_o)_{d_{af}}$, $d(q,q_o)_{d_{fl1}}$ and $d(q,q_o)_{d_{fr1}}$ terms express the distance measurements taken from front, front-left which is between front and front-left-diagonal and front-right which is between front and front-right-diagonal. Repulsion gain 'η' is increased (4.53) as approaching to the obstacle (d_o distance) in configuration space. The distance from robot to target is $d(q,q_h)$. Similarly, attraction gain 'ζ' is increased (4.54) as approaching to the target (d_t distance) in configuration space. Distance sensors are $n=\{1,2,\dots,pn\}$ around the robot and measurement taken by them are expressed with $d(q,q_o)_{d_1}, d(q,q_o)_{d_2},\dots,d(q,q_o)_{d_{pn}}$. The average value of the total measurement taken from sensors around the robot is expressed with $d(q,q_o)_{d_{at}}$. The total number of distance preceptor is pn. Potential calculation order factor 'k' (or $d*_h$) value which is a critical parameter is also raised (4.55) with small rates as

approaching to the obstacle (d_o distance). It has been changed between the 2.85-3.20 value ranges. The 'γ', 'δ' and 'ε' parameters are positive constant factors. The minimum attractive potential threshold 'ap_{min}' in any point on the entire configuration space is assigned to partly overcome the local minima issue. This minimum potential force value is decreased (4.56) dynamically with small rates as approaching to the target (d_t distance). 'ϑ' parameter is also a constant scaling factor.

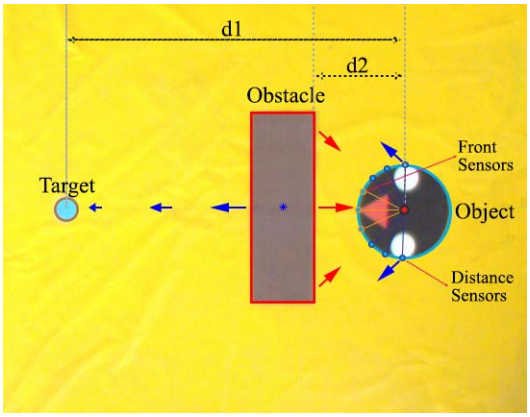

Fig. 20. Objects and variables in working environment

$$d(q,q_o)_{d_{af}} = avg\left(d(q,q_o)_{d_f} + d(q,q_o)_{d_{fl1}} + d(q,q_o)_{d_{fr1}}\right) \tag{4.52}$$

$$\eta = \eta + \eta * \gamma\left(\frac{1}{\sqrt{d(q,q_o)_{d_{af}}}}\right) \Rightarrow d_o \downarrow \tag{4.53}$$

$$\zeta = \zeta + \zeta * \delta\left(\frac{1}{\sqrt{d(q,q_h)}}\right) \Rightarrow d_t \downarrow \tag{4.54}$$

$$k = k + k * \varepsilon\left(\frac{1}{\sqrt{\frac{\sum_{n=1}^{pn} d(q,q_o)_{d_n}}{pn}}}\right) \Rightarrow d_o \downarrow \tag{4.55}$$

$$ap_{min} = ap_{min} + ap_{min} * \vartheta\left(\sqrt[4]{d(q,q_h)}\right) \Rightarrow d_t \downarrow \tag{4.56}$$

Some key points detected between obstacles are generally accepted as node in graph-based methods. Path planning is implemented on these nodes. Therefore, finding these key points (nodes) is an additional task besides path planning. However, in APF method; total resultant force is found by taking resultant of total repulsion

force created by obstacles and total attraction force created by the target on the robot. This vector force provides readjusting the position needed to be arrived, velocity and direction parameters in every step.

Before the main experiments, to show advantages of the A-APF; default and adaptive APF methods have been executed on two different configurations. The simulation results of these pre-experiments have been demonstrated in Fig. 21. Default APF has reached to the solution in 972 steps (or frames) for conf-a, on the other hand, A-APF has reached to the solution in 76 steps. For second configuration conf-b, default APF has reached to the solution in 290 steps, whereas A-APF has reached to the solution in 64 steps. It is clearly seen that A-APF is far more superior in terms of performance.

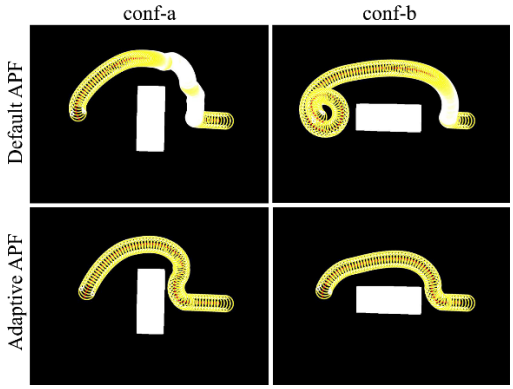

Fig. 21. Simulation instances for several configurations

Table 1 demonstrates the path cost for default and adaptive methods. A-APF has provided more efficient path extraction compare to default one. Although default APF seems to be extract smooth looking path for conf-a, there are excessive amount of unstable oscillations. It completely confused the path trajectory for conf-b. In addition, A-APF has completed the given configuration in less time,

Table 2. These excessive time periods mainly stemming from oscillations; so, from the constant parameters. Ultimately, A-APF has overcome these issues, since it continually updates its parameters in each iteration.

Table 1. Path costs

Method	***conf-a***	***conf-b***
D-APF	625.60px	1015.89px
A-APF	568.98px	559.73px

Table 2. Path extraction times

Method	*conf-a*	*conf-b*
D-APF	*315.237s*	*129.622s*
A-APF	*8.236s*	*7.407s*

Fig. 22 shows (a) k – potential calculation order factor 'K', ap_{min} – minimum attractive potential 'MAP' and (b) η – repulsive potential scaling 'RPS' and ζ – attractive potential scaling 'APS' parameter changes for conf-a configuration. Similarly, Fig. 23 shows these parameter changes for conf-b configuration. k parameter has been shown with blue color and ap_{min} has been shown with red color in (a). η parameter has been shown with red color and ζ parameter has been shown with blue color in (b). Parameter changes are more evident at some points where robot approaches to an obstacle or to the target. After passing an obstacle, it can be seen that the parameters changes in a smoother way.

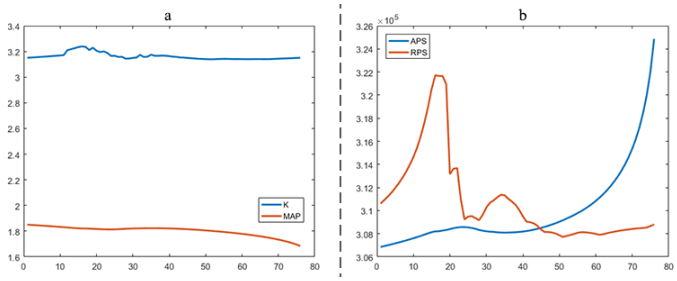

Fig. 22. A-APF parameter changes – conf-a

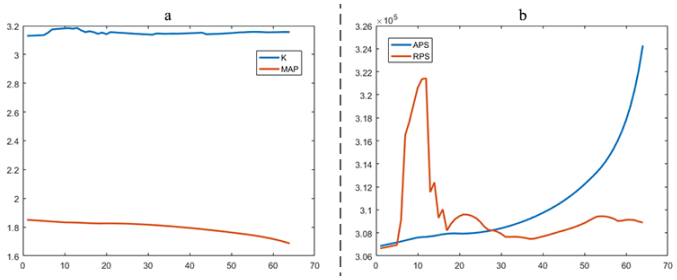

Fig. 23. A-APF parameter changes – conf-b

4.2.4. Decision Tree and Visual Based Control

Decision tree (DT) is a tree-like graph that represents model of decisions and generate new decisions according to possible inputs. It is used to create a suitable decision by branching new levels until the given input is satisfied. In decision tree,

each internal node expresses a "test" on an attribute, each branch signifies the result of the test and each leaf node corresponds label of a class. The paths from root to leaf means classification rules. It is also used to model expert systems as well. It is simple, flexible and provides fast outcome acquiring.

Decision tree is used to model a specialized controller for VBC. It determines WMR control parameters by using available input parameter(s) in this study. Decision tree structure produces suitable velocity parameters in control process according to angle values of triangle. The structure of the tree to generate control parameters is illustrated in Fig. 24. After calculating control parameters, velocity assignment is performed according to decision tree shown in Fig. 25. This second decision tree takes difference of angle values on the mobile robot wheels.

A_d value is the absolute value of difference between A_L and A_R angle values. Branching from this node to below tree layer is made according to the predefined value ranges. For example; assume that A_d value is '8^o', then branching actualizes from the first node to the leftmost node in the second layer. The third layer contains α and β values that are coefficient parameters. As the angle difference increases the difference between these coefficients also increases. Tree branches include branching conditions.

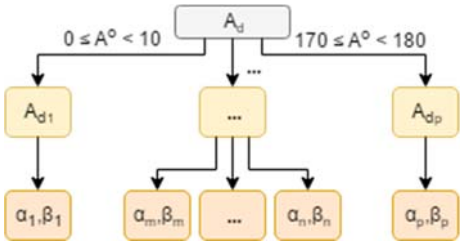

Fig. 24. Angle difference – Control parameters decision tree

Equation (4.57) express general branching condition. α and β parameters take new values (α_n and β_n) according to A_d value within the defined range. If A_d does not match to the defined range, it is controlled whether it matches or not to next range condition (NRC) and this process proceeds in this way.

$$A_n < |A_d| < A_m \, ? \, \alpha = \alpha_n, \beta = \beta_n : \alpha, \beta = NRC \tag{4.57}$$

After gaining the α and β control parameters second decision tree takes sign value of A_d and acquired control parameters to calculate required velocities for mobile robot wheels. The A_v parameter represents sign or magnitude of the difference value.

There are three possible signs '−, 0, +'; so, decision tree has three nodes in second layer. If the sign is negative, branching occurs to the left and V_L and V_R calculated with -/+ additional values respectively. If the sign is positive, then V_L and V_R calculated with -/+ additional values. If the sign is neutral/zero, then V_L and V_R calculated with same +/+ additional values.

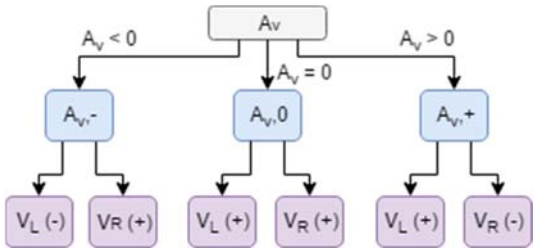

Fig. 25. Angle magnitude – Velocity assignment decision tree

Velocity parameters of WMR are characterized by using following equations. In equation (4.58) V_{max} is the maximum velocity limit that the mobile robot can achieve. V_c parameter is a constant velocity coefficient parameter. The Δ_T is the first distance value between WMR and target positions. The first distance value means that the distance calculated in initial starting configuration space. The Δ_{T_n} is the new distance value between WMR and target positions in configuration space. It means that the actual distance value after starting mobile robot motion. The A_A parameter in (4.59) is the average value of A_L and A_R sum. The γ in (4.60) is the average value of total scaling factors. The V_L represents the velocity of left wheel and the V_R represents the velocity of right wheel. The A_L, A_R and A_T are the interior angle values of the triangle structure as indicated before. The α and β are the constant scaling factors. The A_d parameter is the absolute value of difference between A_L and A_R. Velocity values are calculated by using (4.61), (4.62) and (4.63) equations. Velocity computation of a wheel is performed by depending on opposite corner angle value according to wheel position. The scaling factors α and β affects the velocity magnitude with different rates as angle values change. This dynamic parameter changing operation provides smoother motions instead of sharp motions for WMR.

$$V_{max} = V_c + V_c * \frac{\Delta_{T_n}}{\Delta_T} = V_c \left(\frac{\Delta_T + \Delta_{T_n}}{\Delta_T} \right) \tag{4.58}$$

$$A_A = \frac{A_L + A_R}{2} \tag{4.59}$$

$$\gamma = \frac{\alpha + \beta}{2} \tag{4.60}$$

$$V_L = (V_{max} + A_R) * \alpha \mp \sqrt{(A_d + 1) * \frac{1}{A_T + 1}} \tag{4.61}$$

$$V_R = (V_{max} + A_L) * \beta \mp \sqrt{(A_d + 1) * \frac{1}{A_T + 1}} \tag{4.62}$$

$$V_{R,L} = (V_{max} + A_A) * \gamma + \sqrt{(A_d + 1) * \frac{1}{A_T + 1}} \tag{4.63}$$

A. Calculation of Path Cost Difference

D_S and D_A path is used to calculate the proportion of the path cost between the simulation path and real path formed by robot motions until arriving the target position. Path cost difference (PDC) is simply computed by employing (4.64). This value provides to see how much difference occurs between these paths. In this equation, D_S corresponds to distance of simulated path acquired from path planning process and D_A stands for distance of real path shaped by WMR.

$$PDC \rightarrow \frac{D_S}{D_A} = \frac{\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}}{\sum_{x_1 y_1}^{x_n y_n} \sqrt{\sum_{j=1}^{m}(q_j - p_j)^2}} \tag{4.64}$$

B. Procedures of Stopping Process for Kinematics

Stopping procedure is performed when a predefined threshold parameter is satisfied. This threshold parameter is the angle value between edges at the target position (A_T) for triangle approach. When A_T value reaches to or exceeds the '60^o' by increasing gradually, the '0' pulse values are transmitted from $V_{L,R}$ calculation to the wheels with parameter changing to stop the robot. According to current A_T value, robot velocity is decreased in each iteration step as approaching to the target location, progressively. A_T is controlled for each image frame in real-time. $V_{L,R}$ takes value '0' when the target position is reached. The stopping process of robot starts with (4.65) for this control infrastructure.

$$V_L = 0 \;\&\; V_R = 0 \; iff \; \mathrm{A_T} \geq 60 \tag{4.65}$$

If this condition is not satisfied, then the velocity calculation is proceeded as it has been done previously in (4.61), (4.62) and (4.63). Calculated speed values are converted to the tick/pulse type with required intermediate processes and then the values are transmitted to the robot with ordinary procedures.

C. Control Scheme for Visual Control System

The entire working scheme for designed control model is demonstrated in Fig. 26. In each control process, an image frame is captured from fixed head camera in real time. Object detection task has been executed on this image frame. A path plan is extracted by using adaptive potential field method (A-APF). This path plan is created at first control loop. After this first step, same path plan is used throughout the all control process. According to the used model approach a triangle structure is formed by utilizing detected objects. This triangle is formed between wheel labels and a middle-point located on the extracted path. This middle-point is selected by using a pre-defined skip-factor in path matrix. For triangle model degree values (angles) and of edge intersections are calculated. Decision tree function is employed by getting 'x' difference (A_L-A_R) as input value. Result of this function is added to/subtracted from wheel velocity values according to sign of this 'x' value. In velocity computation step, the signs $(+,-)$ express that velocity of left wheel increased and right wheel is decreased by decision tree. Similarly, $(-,+)$ express vice versa. In addition, $(\sim,\sim)$ expresses that the velocities of both wheels are influenced with the same rate. Ultimately, threshold ($A_T \geq 60$) is controlled. The control process is stopped, if condition is met. Otherwise, control loop proceeds from initial step with next frame.

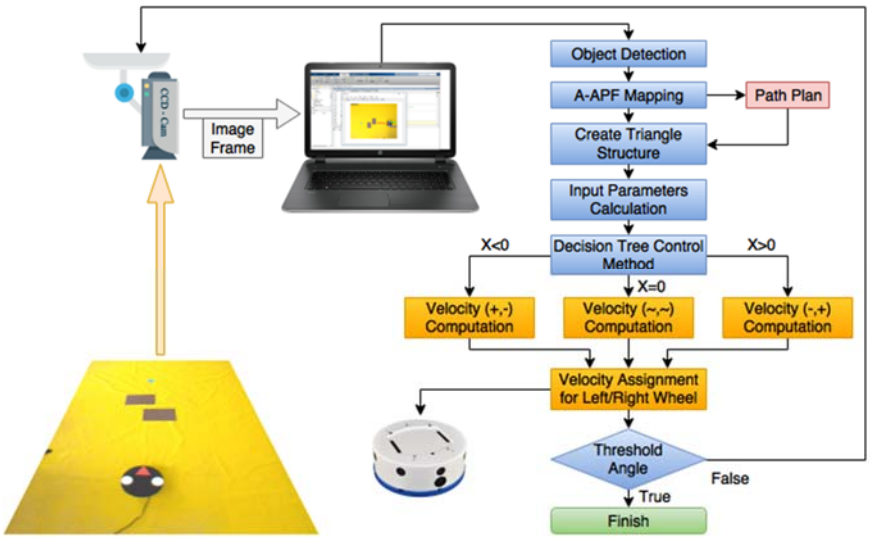

Fig. 26. General working phases for designed control method

Using Cartesian coordinates on detected objects and path plan; a triangular positioning scheme is created to generate the control input. This triangle is formed between the wheel labels and a threshold point (intermediate target) on the obtained path. The inputs obtained from this scheme are given to the decision tree controller and the WMR velocity parameters are calculated. The threshold is selected using a predefined jump factor in the path matrix. In each iteration, the threshold point is updated according to the jump factor value, depending on the angle A_T at the target. The update process of this threshold point ends when the final target which is the last position of the path is selected. Fig. 27 shows the threshold point representation on a path plan. These threshold points can be defined as intermediate targets.

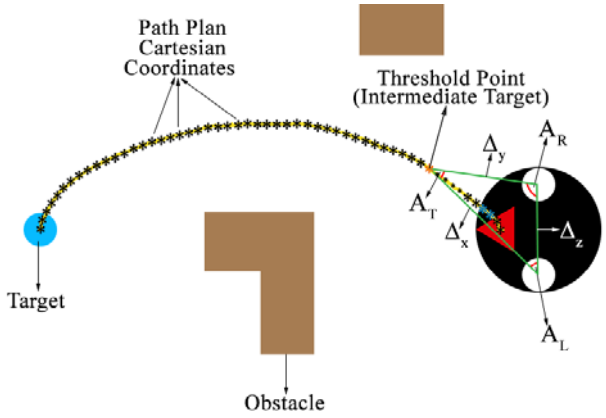

Fig. 27. Path plan threshold point representation

The operating layers of the whole system is demonstrated in Fig. 28. Image frame is taken from the fixed head imaging device (a), threshold process is carried out (b), object detection and geometric parameters are calculated (c), taking local frame around the robot within defined threshold value (d), according to object positions simulate and extract a path plan with A-APF (e), calculate input parameters according to path plan and robot position (f), velocity parameters for both wheels are calculated (g), convert and send parameters to the physical mobile robot (h).

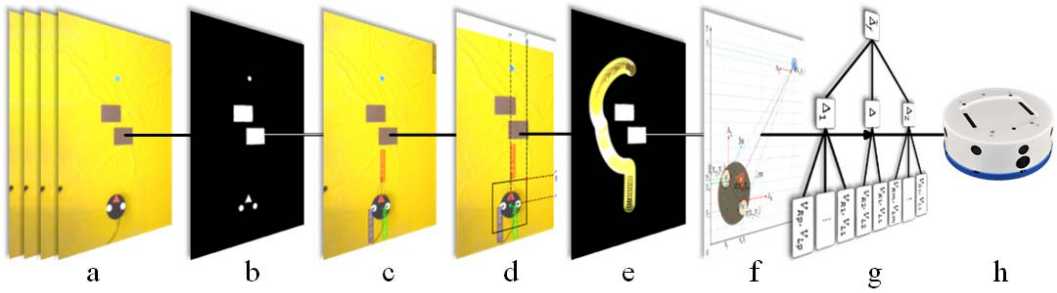

Fig. 28. Operating layers of the designed system

4.3. STAGE-3: Multi-Camera Extension

The multi-camera configured control system has four sub-stages: In first stage, physical environment including configuration space, camera positions and connection/cabling are prepared. In second stage, image-based tasks; image stitching and object detection are implemented, then environment map is created. In third stage, path plan is extracted and divided. In last stage, the WMR motion control is implemented. The stages are illustrated in Fig. 29. The system modules and configuration space are given in Fig. 30.

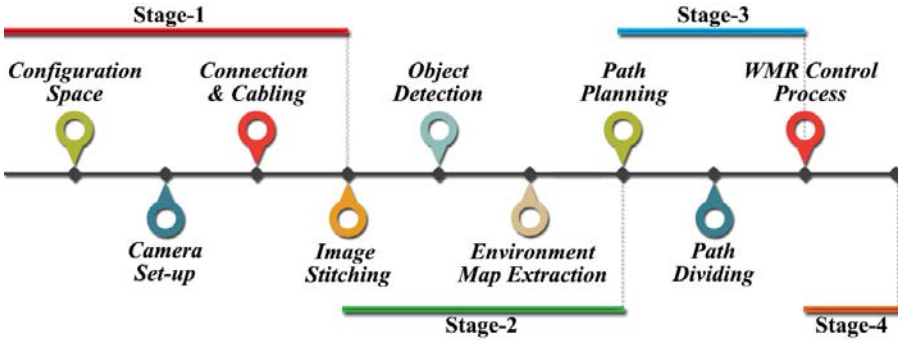

Fig. 29. Stages for the multi-camera control model

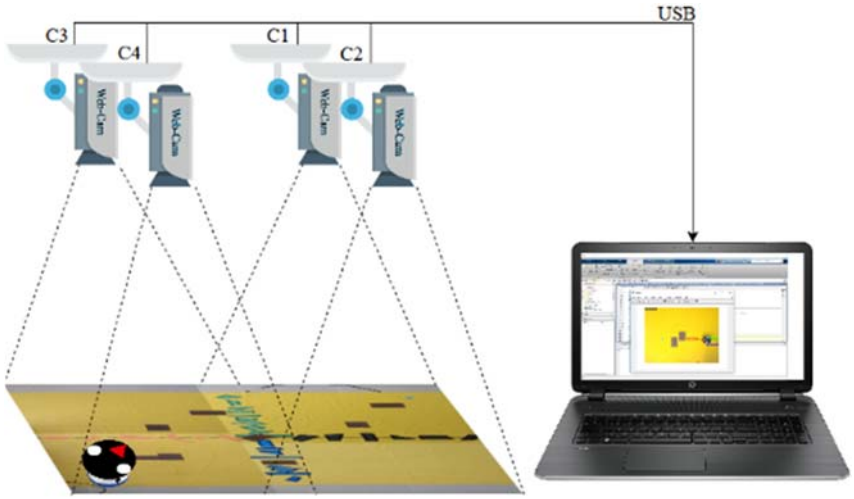

Fig. 30. System modules and configuration space

4.3.1. Operating environment of Stage-3

Operating environment includes all Stage-2 properties. Besides, configuration space observed through four cameras that have same specifications. All cameras are placed a fixed position with same heights from the ground. They have been hanged 210 cm above from the floor. Their lenses are perpendicular to the floor. The cameras are connected to the computer with USB 2.0 ports. Each camera covers an area

including two axes labels according to camera position. Experiments have been performed on Intel i3-3217U 1.80 GHz CPU with 6GB DDR3 1600 MHz Memory and 5400 Rpm HDD. The utilized mobile robot steering wheels are quite thin and ball caster wheels are small. Therefore, plastic yellow layer is used to prevent wheel jamming to suture area of the floor tiles. The brown rectangle objects are obstacles and blue circular object is the main target. Different colored and shaped labels have been placed to the floor as distinctive properties for feature detector. Fig. 31 demonstrates working space representatively. Cameras are expressed with C1, C2, C3 and C4. Viewing area for C1 is the multiplication of C1_x and C1_y. Intersections of two camera viewing areas are C1-C2, C1-C3, C2-C4 and C3-C4. There is one more intersection area where all camera can see and it is C1-C2-C3-C4. This means that there are several areas tracked by different cameras. The remaining areas are observed with only one camera in environment. Blue objects (T1, T2) represent the target position. Red objects (I1, I2) represent boundary positions where robot starts to enter viewing area of two cameras at the same time on the path. Rp1, Rp2 and Rp3 are reference points between two obstacles. They are used to show path skeleton which resembles the shortest path between given two positions.

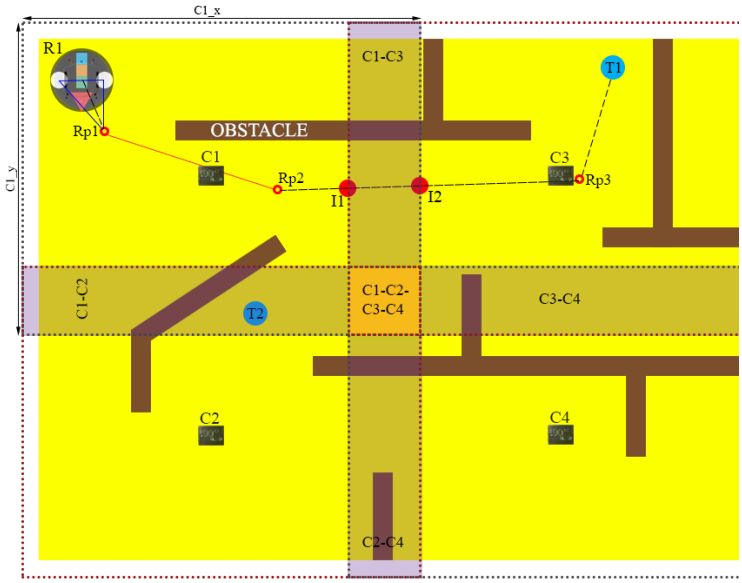

Fig. 31. Working environment for designed control system (Representative)

4.3.2. Image Stitching

Image Stitching is one of the particular studying fields that is commonly researched. It hosts a number of problems needed to be overcome. Generally, there are two basic goals; overlapping the images taken from same position and different

angles, on top of each other and fusing common intersection points in the best possible way. Basic matters needed to be considered in image stitching are as following;

- Density of variables across the whole scene
- Variable density and contrast values between frames
- Lens distortion
    - Pin cushion, Barrel/Bucket and Fisheye
    - Setting the lens profile at the selected focal length
    - Use of available lens profiles
- Dynamics / movements in the scene
    - Shadowing / Ghosting
    - When the images are aligned, basically one of them is selected
- Alignment error (Axis misalignment)
    - Shadowing / ghosting again
    - Better control points should be selected
- Visually satisfactory results
    - Super wide panoramas may not always be satisfactory
    - Gold ratio, 10: 3 or other satisfactory scale trimming

The main problem with image stitching is the difference in the component size between the x and x' regions due to the angle difference as shown in Fig. 32. In the equations (4.66), (4.67), (4.68) and (4.69) below, the components to which x is connected are given [100];

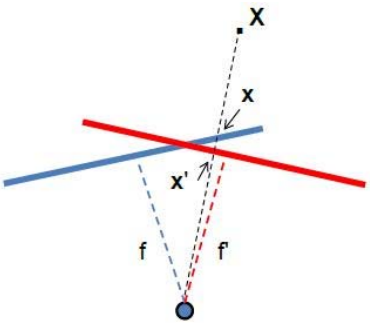

Fig. 32. Parameter changes due to panoramic shooting angle

$$x = K[rt]X \tag{4.66}$$

$$x' = K'[r't']X' \tag{4.67}$$

$$t = t' = 0 \tag{4.68}$$

$$H = K'R'R^{-1}K^{-1} \text{ to be, } x' = Hx \tag{4.69}$$

Typically, only R and f (4 parameters) will change, but usually there are 8 parameters of H (homography). In there, the K and K' are the measurement (calibration) matrices. The X is the actual location of the object. The x and x' represent the position of the same objects taken from different angles at the same focal length. The R and R' are rotation matrices. The t and t' are translation matrices. Fig. 33 shows the distance difference between the components in the region originating from the camera angle with the same red dot common to both images. A component, which is only the second image, is indicated by a green dot.

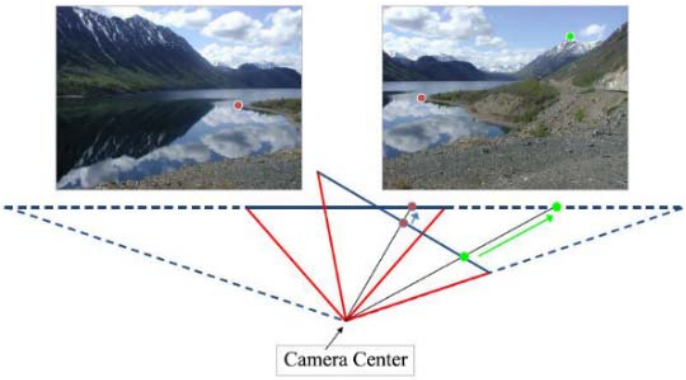

Fig. 33. Images taken from a camera made return motion (Photo: Russell J. Hewett)

The 'n' images will be taken from the 'n' head cameras for image stitching. These images are placed horizontally or vertically relative to the camera positions by superimposing common areas with the next intersection of the image. Although it is similar to creating a panoramic image, it is different from each other in terms of the location in which the image is taken. Images obtained for a panoramic image are taken at different angles with a single camera from the same point. On the other hand, images are obtained from different points but from the same angle (perpendicular to the surface) in the multi camera configuration. Generally, creating a panoramic image with source images taken from the same spot is more prone to distortions in the image. This is the difficulty of matching the intersection points of the images because of the fact that the input source images are taken from different angles. If the feature matchings at these intersections are not sufficient, the stitching success at the relevant region will be low and visible distortions will occur. The presence of common intersection areas closest to each other due to the shooting angle is an enhancement

factor in the images taken consecutively from the central view. Fig. 34 shows two image frames superimposed on top of each other.

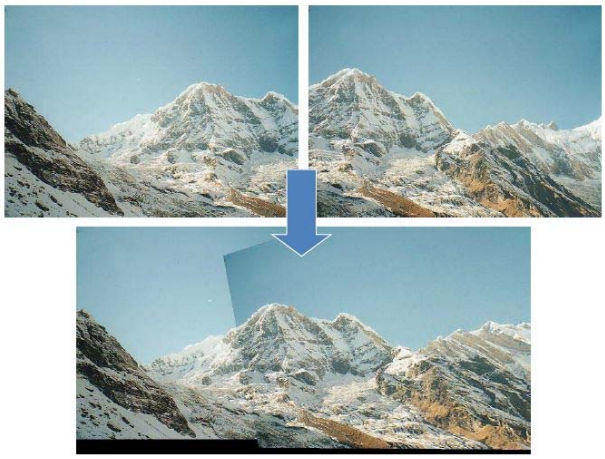

Fig. 34. Two superimposed images

In multi head-cameras, the performance will be higher because the images are taken from the same angle. Since the matching ratio of inter-view intersections is very high, the panorama can be started from the desired image. The key-points can be detected with SURF [101] or SIFT [102] feature detectors to extract image properties. This work will use SURF. The pseudo-code of the image stitching is given in Table 3.

Table 3. Image stitching process

1. Take two images as parameters, *G (n)* and *G (n-1)*
2. Make feature extraction on both images with SURF
3. Calculate the set of matching points (Feature-Match)
4. Apply RANSAC to estimate a homography transforming the image that overlaps the spots, *T (n)*
5. Convert images using this homography
6. Stitch the images together
7. Repeat the process steps to stitch the next image with these blended images

Image properties are detected and matched from $G(n)$ to $G(n-1)$ – (common intersection regions are determined). The SURF features are extracted from the black-and-white form of the first image. Because the images are close enough to the camera, a projective conversion is used. If pictures are farther away, affine transformation is used. Then, in the iteration, the SURF properties of the $G(n)$ image are extracted. Matching of these extracted properties between $G(n)$ and $G(n-1)$ is made. The geometric transformation of $T(n)$ mapped from $G(n)$ to $G(n-1)$ is calculated by the

RANSAC method using property mappings and taking the previous property mappings as parameters.

Transformations mapped from $G(n)$ into panaroma image/view $T(1) * \cdots * T(n-1) * T(n)$ are calculated. It is obtained by multiplying itself and the previous transform together. Moving from the situation that "the center of the captured scene exhibits the least distortion", a good panorama can be obtained by changing the transformations. The change process is done by inverting the transform for the central image and applying this transform to all the other. This case can be neglected in this study, since multiple head-cameras all receive images from the same vertical angle and from different positions. Therefore, angle-induced distortions hardly ever occur. Similar to the previous stages; the object detection process is performed with color segmentation and quantization.

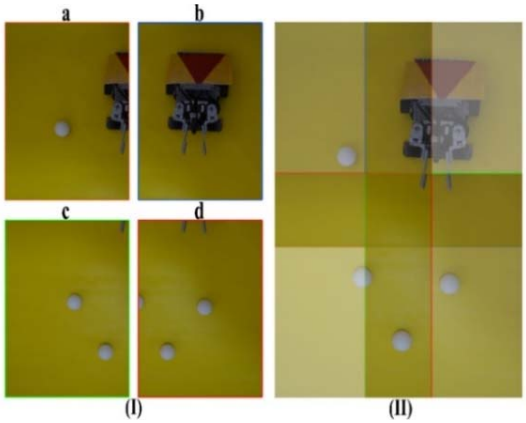

Fig. 35. (I) Images obtained at the same angle from different camera positions (II) stitched state of four-images

In Fig. 35, four images (a, b, c, d) taken from the cameras are shown. The cameras have same specifications. The images taken from different positions and same angles are stitched on common intersection points, (I). The opacity values have been changed so that the stitched areas in the images look clear, (II).

4.3.3. Robot Control in Multi-Camera Configuration

All designed controllers and positioning models are generally used with a fixed single head-camera configuration in the literature. In this study, number of cameras is increased to create an expandable/scalable configuration space. Four cameras are used to acquire image frames from configuration space as shown in Fig. 36. These acquired images are stitched and fused to get one bigger image. After this stage, the object detection is performed on this fused image, so that the binary map is obtained. Next

step, the path plan is extracted from the binary map with A-APF (Adaptive Artificial Potential Field). Then the controller method drives mobile robot throughout this path in real time until it reaches to the target position.

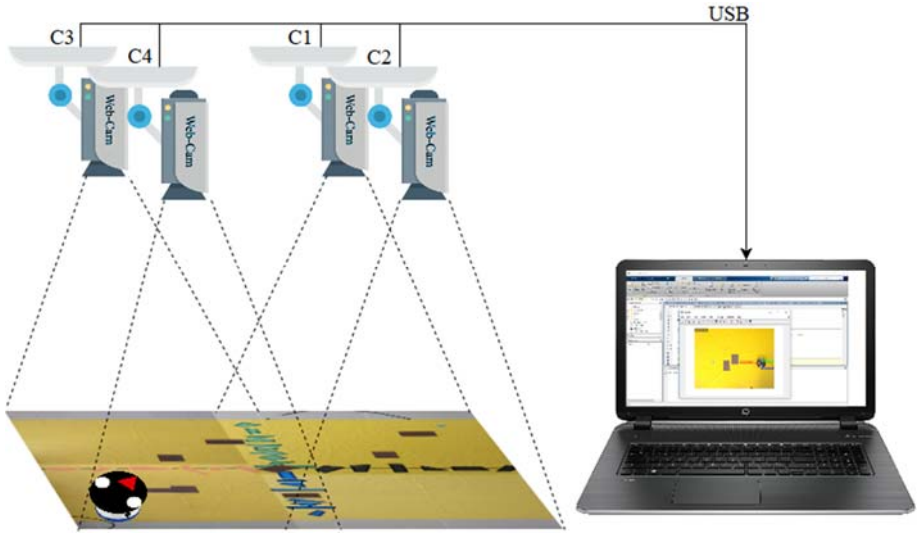

Fig. 36. Multi-camera – Computer connection and configuration space

Acquired path from the stitched image is expressed with the S_{path}. This path variable contains coordinates of the path which are obtained from the A-APF path planning process. In the next stage, S_{path} is divided (4.70) according to each related camera viewing area (CVA). The $S_{x_{path}}$ represents the path fragment under the camera x (C_x) and it is called sub-path. The S_{path} can be divided to one sub-path, several sub-paths or maximum four sub-paths. The S_{path} does not be divided if the mobile robot, target are under the same camera and target is reachable from there. The divided paths assigned to a path tracking queue (P_{tq}) according to the path tracking order (P_{to}). For instance, if the S_{path} is pass through C_2 and C_4 then it divided as $S_{2_{path}}$ and $S_{4_{path}}$ and these sub-paths are added to P_{tq} with $P_{to} = (2,4)$. This means that the mobile robot reaches to the target by using C_2 and C_4 cameras. Sub-paths have one sub start point ($S_{x_{SP}}$) and one sub target point ($S_{x_{TP}}$). For instance, the mobile robot starts to move from a $S_{2_{SP}}$ towards $S_{2_{TP}}$ under the C_2 CVA. When it reaches to the $S_{x_{TP}}$, its current position is assigned as $S_{x_{SP}}$ of the next path in P_{tq}. This iterative process continues until the all sub-paths in P_{tq} are processed.

The S_{path} coordinates are distributed to the path hosted CVAs with rescaling process by using (4.71). x_{new} and y_{new} are the new (x, y) path coordinates that are

desired to calculate in undistorted image space. X_{und} and Y_{und} are the width and height of the undistorted image. x_{old} and y_{old} are the path coordinates in stitched image. X_{dis} and Y_{dis} are the width and height of the distorted image. To detect whether the robot appeared in the intersection area or not; this area on the path is controlled by using (4.72) and (4.73) conditional statements. I_{nm} is the intersection area; it takes 1 or 0 value; 1 value means that robot is in I_{nm} intersection area. C_{act} is the active camera. If robot is not in the intersection area then C_{act} is equal to the currently working camera (C_{pre}). I_{all} represents the intersection area of all four cameras. The t_{x_i,y_i} is the main target position in 2D image space. If robot is detected under I_{all}, then the C_x, which holds t_{x_i,y_i} in its CVA, is activated.

$$S_{path} \Rightarrow P_{tq} = \left(S_{x(1)_{path}}, S_{x(2)_{path}}, \dots\right) \wedge \forall x(m) = n, 1 \leq n \leq 4 \wedge n \in \mathbb{Z} \quad (4.70)$$

$$x_{new} = \frac{x_{old} * X_{und}}{X_{dis}} \wedge y_{new} = \frac{y_{old} * Y_{und}}{Y_{dis}}, \forall (x_{old}, y_{old}) \in S_{path} \quad (4.71)$$

$$I_{nm} = 1 \; ? \; C_{act} = C_x : C_{act} = C_{pre} \quad (4.72)$$

$$I_{all} = 1 \; ? \; C_{act} = C_n \Leftrightarrow \exists \, t_{x_i,y_i} \in C_n : C_{act} = C_{pre} \quad (4.73)$$

The $S_{x_{path}}$ is tracked according to its path coordinates. Firstly, a sub-target $S_{x_{st}}$ is determined in these coordinates according to a pre-defined threshold. In each control iteration, the next path coordinate which is after threshold position is assigned to the $S_{x_{st}}$. In each $S_{x_{path}}$, $S_{x_{st}}$ starts from threshold position according to $S_{x_{SP}}$ and updated continually until $S_{x_{TP}}$ is assigned to $S_{x_{st}}$. When the robot reaches the last $S_{x_{st}}$ the next C_x takeover the tracking process for related $S_{x_{path}}$. As a result, the motion is modeled according to progressively updated $S_{x_{st}}$ sub-target. This process provides smooth motion especially in sharp turns in sub-path plan. Fig. 37 illustrates the sub-target tracking process.

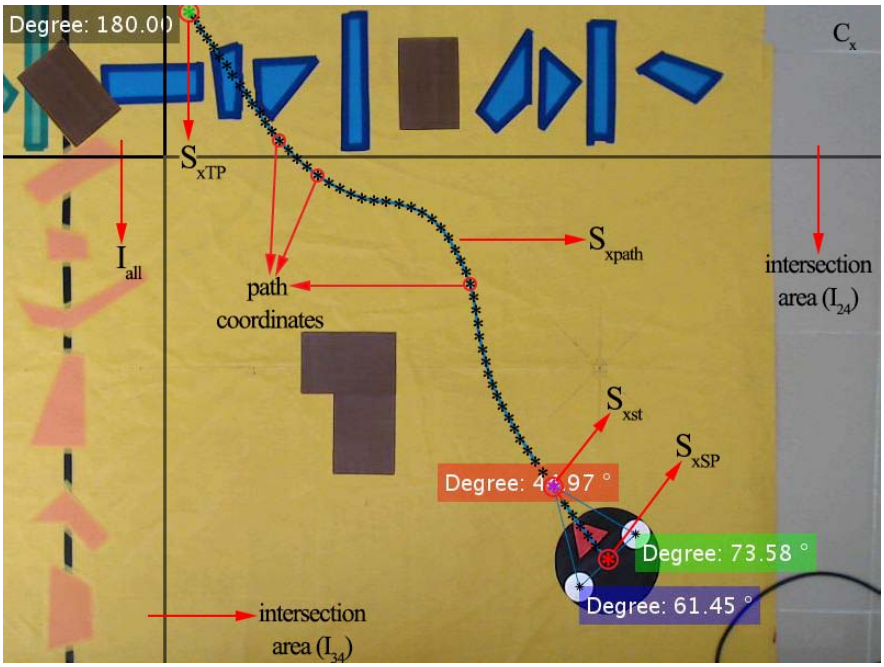

Fig. 37. Sub-path and path tracking under C_x

The entire multi camera-based control process flow is demonstrated in Fig. 38. Unlike the single head-camera configuration, designed infrastructure works a bit different for multi head-camera configuration. Acquired image frames are stitched and object detection is made on this stitched image; so, the binary map of the environment is obtained. A path plan is formed on this map. After acquiring the path plan, path is divided and distributed to the required regions on the map. The robot is tracked with related camera where it is in the field of view. In other words, the overhead camera which can cover the robot starts tracking process. The controller triggers the motion of mobile robot. When robot move towards to the field of view of another camera; firstly, control process is proceeded until robot fully appears in the intersection region. According to related camera(s) of this intersection region, the next head camera is activated with respect to view field of other camera where the next part of extracted path is located. When this second camera is activated, the first camera is deactivated instantly. This means that only one camera is activated at a time. This process continues until the mobile robot reaches to the main target (t_{x_i,y_i}) position. At each intersection point suitable camera take over the tracking process and control task is proceeded in this local configuration space. Suitable camera means that path is in field of view from last intersection region where robot reaches to the next intersection region or target position. The extracted path from whole map can be in one or more camera field of view. This situation changes according to target distance,

obstacle positions and robot position and direction. In each camera region, the controller is executed normally without any additional workload. The controller is only stopped or actuated when there is an intermediate or main target. Summary of the multi camera-based control system is illustrated in Fig. 39.

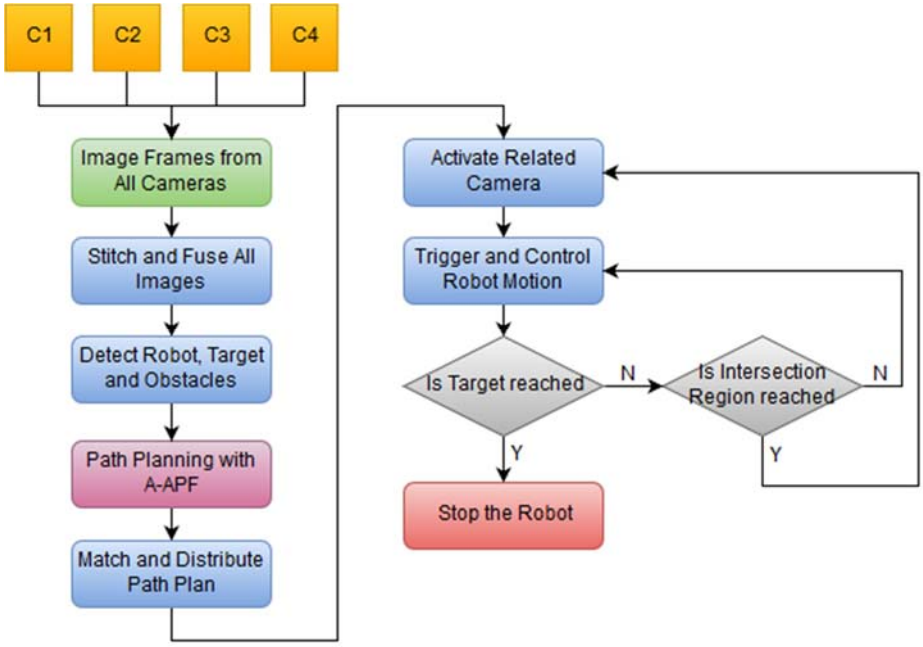

Fig. 38. Multi camera-based control process flow of designed system

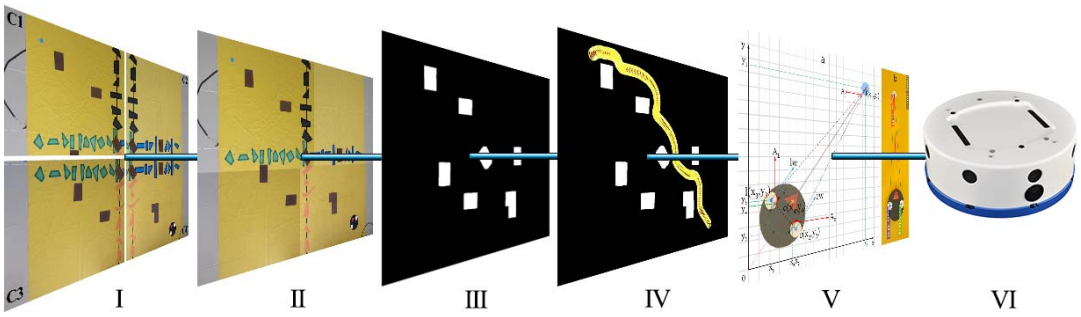

Fig. 39. Summary of the multi camera-based control system: (I) Simultaneously acquired images from all cameras (II) Stitched image (III) Detected obstacles (IV) Extracted path plan between robot and target (V) Calculation of controller inputs (VI) Robot implementation

5. EXPERIMENTS

5.1. STAGE-1: Obstacle Free Experiments

5.1.1. Experiment Configurations

Velocity of wheels are calculated by processing image frames; the input parameters of controller are fluently calculated until the mobile robot arrives to the target position in an indoor environment. The distance from starting to the target positions is set to 100cm in experiments. The S_{max} value is adjusted to '80 tick/ps' in all experiments. The starting direction (heading) of the WMR is set to two different directions; '0°' and '180°' according to the position of target. The direction statuses are adjusted to test position of simple and extreme conditions in both positioning models. Because the angle value between the target and mobile robot changes in the range of minimum 0° to maximum 180°, we have selected and used these two angle directions. There are four different conditions for experiments;

- Distance of 100 cm and angle of 0° implementation for the graph-based model; it is named as ***100_0_G***
- Distance of 100 cm and angle of 180° implementation for the graph-based model; it is named as ***100_180_G***
- Distance of 100 cm and angle of 0° implementation for the triangle-based model; it is named as ***100_0_T***
- Distance of 100 cm and angle of 180° implementation for the triangle-based model; it is named as ***100_180_T***

These four experiment conditions have been implemented three times in the configuration spaces. Therefore, 12 experiments have been performed with three repeats and the average values of the repeated experiments are computed to obtain accurate results for these four experiment conditions. The robot is gradually fitted to a smooth trajectory after a period of motion. The obtained information from control model processes are stored to hard drive for additional comparisons.

5.1.2. Graph Based Control Model

A. Control Model Experiments

In Fig. 40 and Fig. 41; the starting positions of the WMR are shown with (a) frame and finishing positions are shown with (b) frame. The small frames at the bottom demonstrate the robot positions during processing time. It should be said that the black line is the power cord shown behind the robot. The 100_0_G experiment is given in Fig. 40. The red, green, blue and white labels represent the D_1, D_2, D_W and D_T, respectively. The path trajectory followed by the robot is demonstrated with red path in (b) frame. When $D_T \leq D_W - 30$ conditional statement is true, the robot has been stopped. The path trajectory has been fluctuated at some sections. These fluctuations stem from the unsystematic errors. Preventing them is a challenging issue, but the control system successfully manages these errors. The 100_180_G experiment is demonstrated in Fig. 41. According to the target position the robot is placed in the reverse direction. Trajectory is illustrated with the red path in the (b) frame. All initial and final control parameters are given in Table 4.

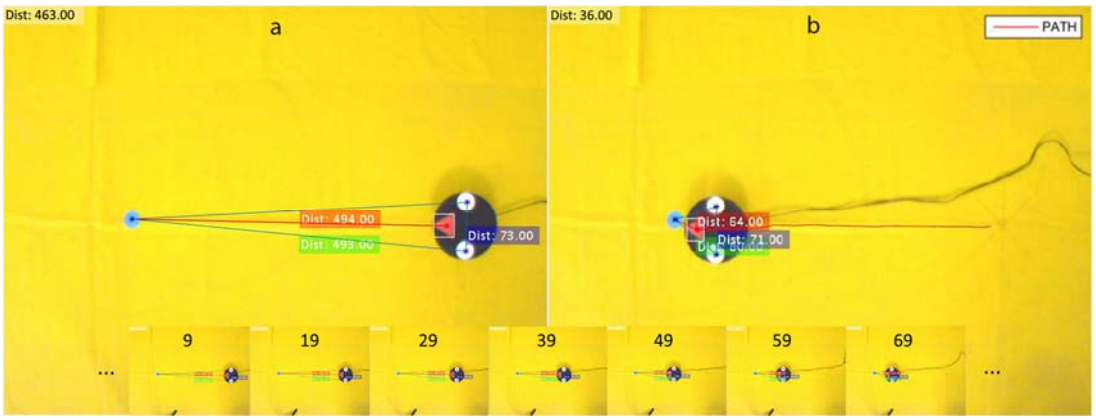

Fig. 40. (a) Starting position of mobile robot (b) Finishing position of mobile robot

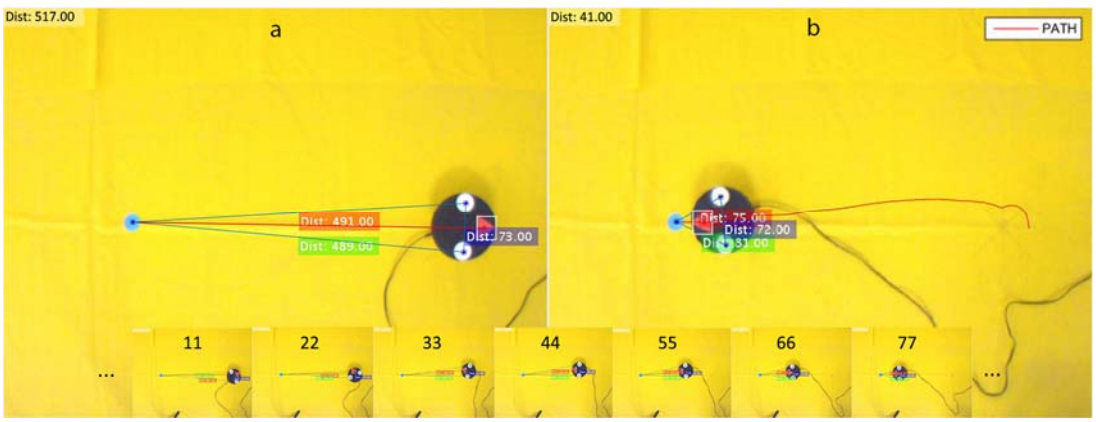

Fig. 41. (a) Starting position of mobile robot (b) Finishing position of mobile robot

Table 4. Graph based control model experiments

Experiment	Initial (px)				Final (px)			
	D_1	D_2	D_W	D_T	D_1	D_1	D_W	D_T
100_0_G	494	493	73	463	64	80	71	36
100_180_G	491	489	72	517	75	81	72	41

B. Control Model Observations

100_0_G Experiment

Fig. 42 (a) represents weight changes for the experiment of 100_0_G. The D_1 and D_2 parameters approximate to each other in some frames. There are several disruptions in distance value by starting 30th frame. The sudden distances disruptions stem from irregularity of floor plane or a foreign object on the path. By starting 62nd frame, the difference between D_1 and D_2 increases within several frames. According to several experiments, the frame loss causes this parameter differentiation.

Fig. 42 (b) demonstrates the velocity variations of robot wheels for the 100_0_G experiment. The f_G function gives sensitive responses to changing in distance until the 55th frame. The WMR has been tried to be kept in the path trajectory, continually. The deviation from the target may be caused by irregularity of the floor. The velocity variables are stable after that point.

The 77 images are processed in 9.08s. So, 8.480 image frames per second are achieved and it takes 0.117s to be processed for each image frame. If storing and displaying is inactive for this experiment; then 14.745 images per second are achieved. These performance rates are admissible for a real time VBC system.

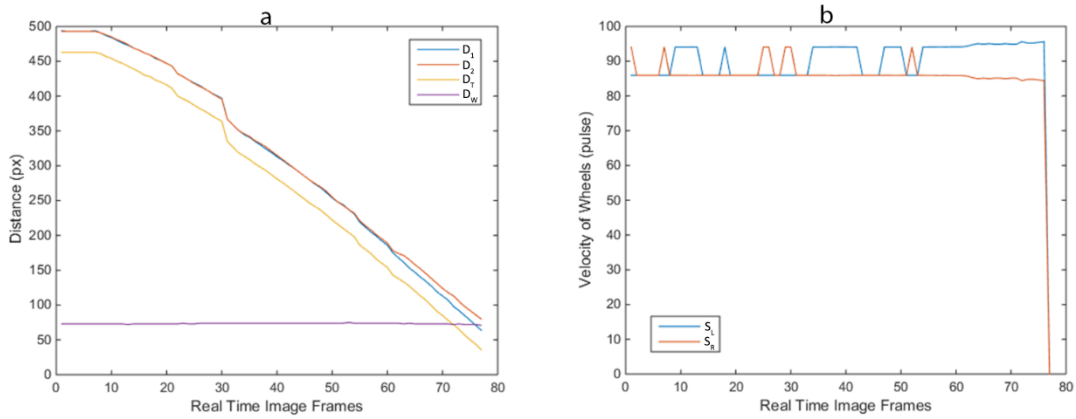

Fig. 42. (a) Distance changes of mobile robot (b) Velocity changes of mobile robot

100_180_G Experiment

Fig. 43 (a) represents weight changes for the experiment of 100_180_G. The parameters D_1 and D_2 are nearly equal in some image frames as from the starting position. The difference between D_1 and D_2 parameters increases until the direction of robot has 90˚ or below value according to the target position after several frames. That motion behavior proceeds to the 20th frame, after this frame; these two-distance values approach to each other. The D_W value changes between 71px and 74px and its average is 72px. The D_T declines stably until the $D_W - 30$ condition is satisfied.

Fig. 43 (b) shows velocity variations of the robot wheels for the 100_180_G experiment. The S_R has drastically increased and the S_L has drastically decreased at 19th frame. The S_L is amplified and S_R is minified until the direction of robot fits to the linear path trajectory towards the target after several frames. The S_L and S_R have changed with low rates suddenly at some points until the last image frame. This means that controller has tried to keep the WMR in the path trajectory.

The 86 images are processed in 10.06s. Therefore, 8.548 image frames per second are achieved and it takes 0.116s to be processed for each image frame. If storing and displaying is inactive for this experiment; then 14.915 images per second can be processed with the system.

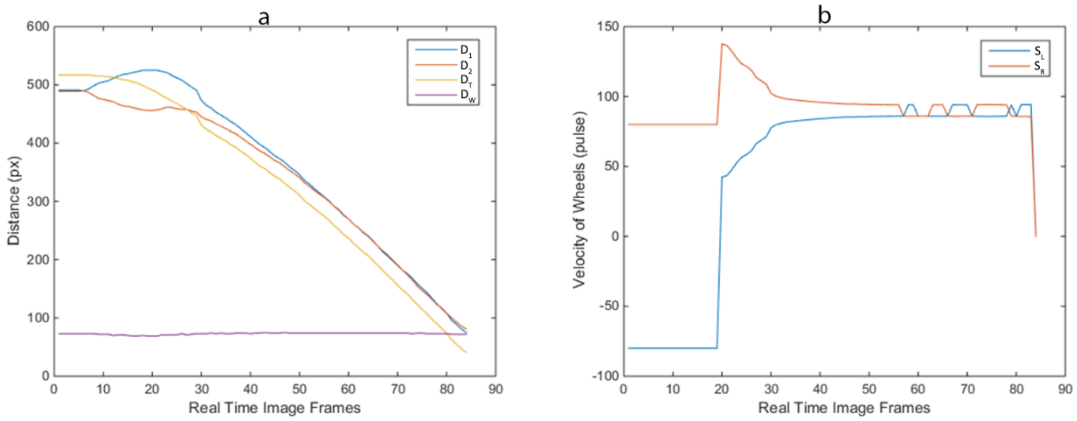

Fig. 43. (a) Distance changes of mobile robot (b) Velocity changes of mobile robot

5.1.3. Triangle Based Control Model

A. Control Model Experiments

In Fig. 44 and Fig. 45; starting positions of the robot are shown with (a) frame and finishing positions are demonstrated with (b) frame. The 100_0_T experiment is

demonstrated in Fig. 44. The blue, green, red and black labels represent the A_L, A_R, A_T and total angle in (a) frame, respectively. The red path represents the trajectory in (b) frame. When $A_T >= 60$ condition is met, the robot motion has been stopped. The path trajectory is emerged almost linearly. The path has been disturbed at some points. The main reasons behind these fluctuations are touched in previous section., The 100_180_T experiment is given in Fig. 45. It implies that according to the target position the robot is placed in the reverse direction. All initial and final values are presented in Table 5.

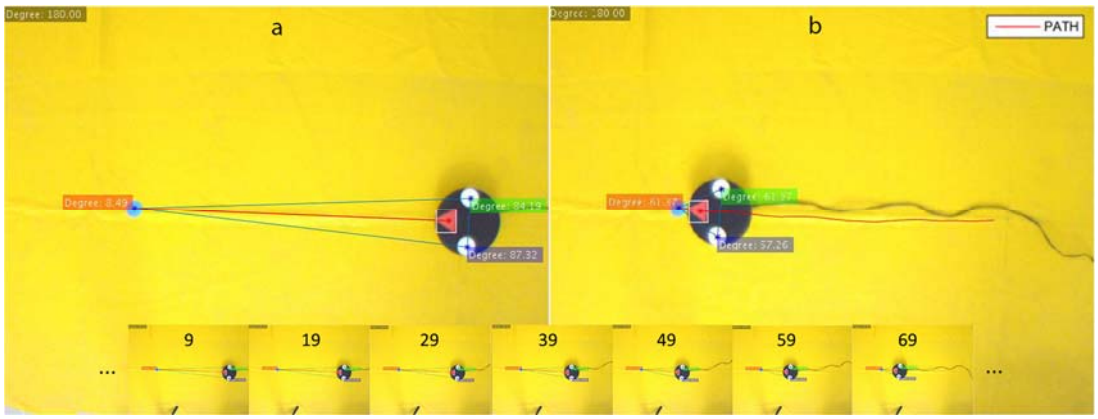

Fig. 44. (a) Starting position of mobile robot (b) Finishing position of mobile robot

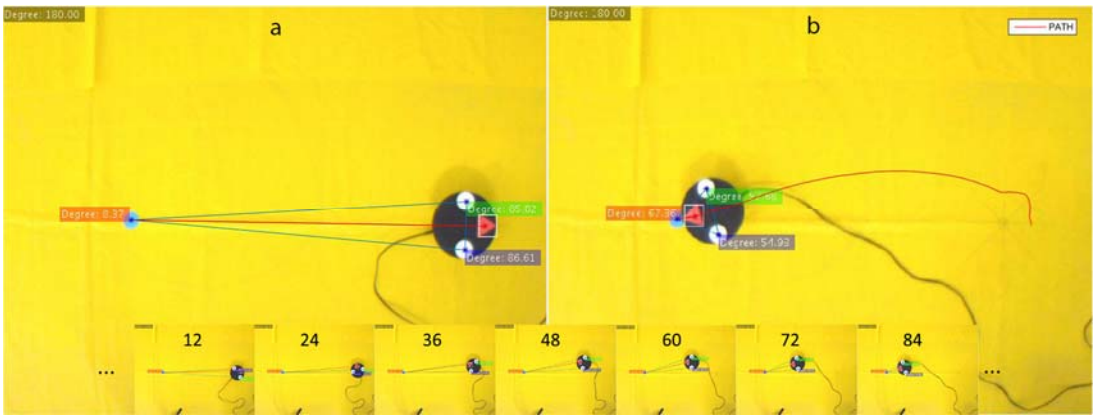

Fig. 45. Starting position of mobile robot (b) Finishing position of mobile robot

Table 5. Triangle based control model experiments

Experiment	Initial (px)			Final (px)		
	A_L	A_R	A_T	A_L	A_R	A_T
100_0_T	87.32˚	84.19˚	8.49˚	57.26˚	61.37˚	61.37˚
100_180_T	86.61˚	85.02˚	8.37˚	54.98˚	57.66˚	67.36˚

B. Control Model Observations

100_0_T Experiment

Fig. 46 (a) demonstrates angle changes for 100_0_T experiment. The A_L and A_R are approximate to each other in several image frames. There are sudden fluctuations in distance value by starting from the 10th frame. If these angle variations are too big, then according to the observations the main reason is several frames loss. The values approximate to each other with small rate of differences by starting the 64th frame. When A_T is equal or greater than 60°, the WMR has been stopped.

Fig. 46 (b) shows changes in velocity of robot wheels for the 100_0_T experiment. The f_G function exhibits sensitive responses to angle variations. The WMR has been precisely tried to be kept in path trajectory towards the position of target. The floor irregularity may give rise to a deflection from the target. But the velocity values are stable except several frame points.

The 77 images are processed in 10.84s. It signifies that 7.103 images per second are processed and it takes 0.140s to be processed for each image frame. If storing and displaying is inactive for this experiment; then 12.236 images per second can be processed with the system.

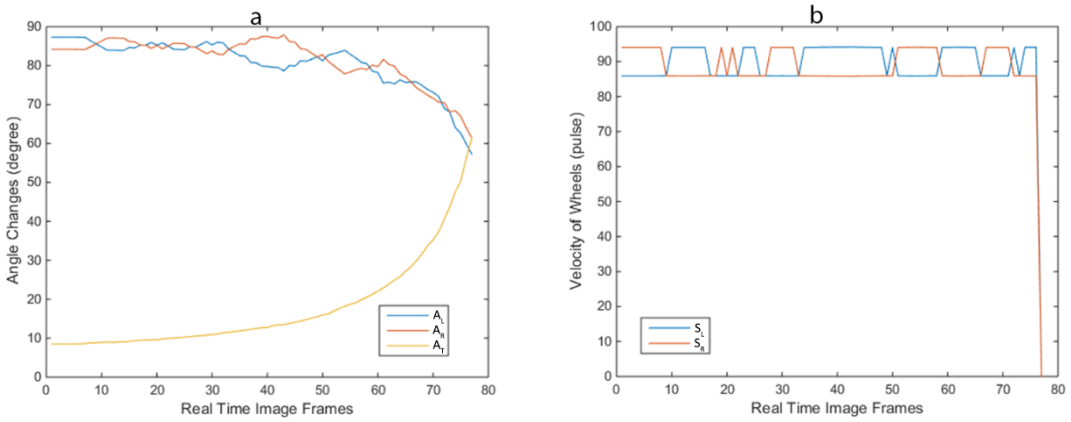

Fig. 46. (a) Angle changes of mobile robot (b) Velocity changes of mobile robot

100_180_T Experiment

Fig. 47 (a) represents angle variations for the 100_180_T experiment. The parameters A_L and A_R values are almost equal in several frames. After several frames, the difference between A_L and A_R increases until the robot direction has 90° or small value according to the target position. That motion characteristic proceeds to the 22nd

frame. These two angle parameters approach to each other after this frame. The A_T parameter decreases to '0' value until the 22nd frame. In later frames, the A_T increases consistently to the $A_T \geq 60$ threshold limit.

Fig. 47 (b) demonstrates changes in velocity of robot wheels for the 100_180_T experiment. The robot performs a rotation motion until 22nd frame where the direction of robot decreases below the 90˚. The S_R has increased drastically, the S_L has decreased drastically at this frame. The S_L is amplified and S_R is minified until the direction of robot fit to the linear path trajectory towards the target position after several frames. The S_L and S_R parameters have changed with small rates in following frames. It means that the controller has tried to keep the robot in fitted path trajectory.

The 92 images are processed in 12.54s. This means that 7.336 images per second are processed and it takes 0.136s to be processed for each image frame. If storing and displaying is inactive for this experiment, then 12.941 images per second can be processed with the system.

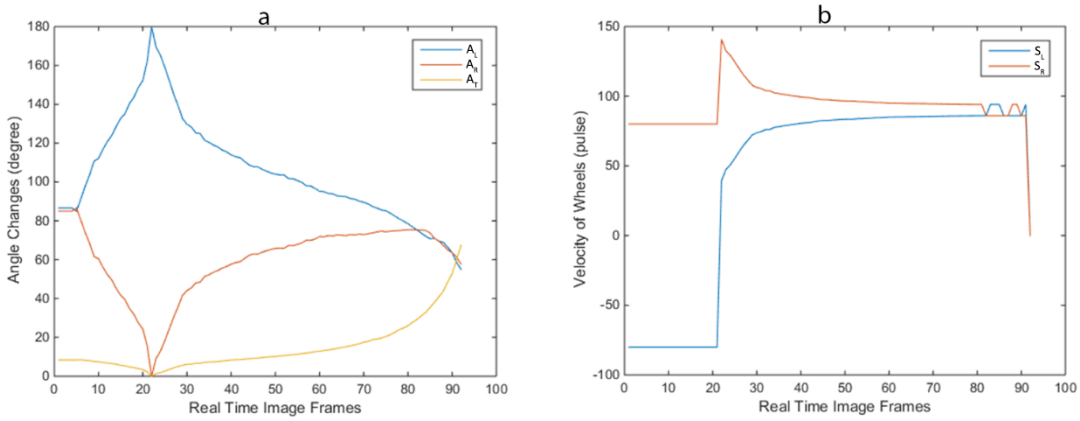

Fig. 47. (a) Angle changes of mobile robot (b) Velocity changes of mobile robot

5.1.4. Additional Experiments

Four additional experiments are conducted except from previous experiments. Two distinct directions as 45˚ and 90˚ have been experimented. They are the starting robot directions. These experiments are shown in Fig. 48 and Fig. 49 for triangle and graph-based positioning models, respectively. The (a1-a2) demonstrates the initial positions and (b1-b2) demonstrates the final positions for the robot in both experiments. The robot has been arrived to the target successfully by the developed controller in both directions. The experiment details are only exhibited as table results. The name additional experiments are coded in the same way as the previous experiments.

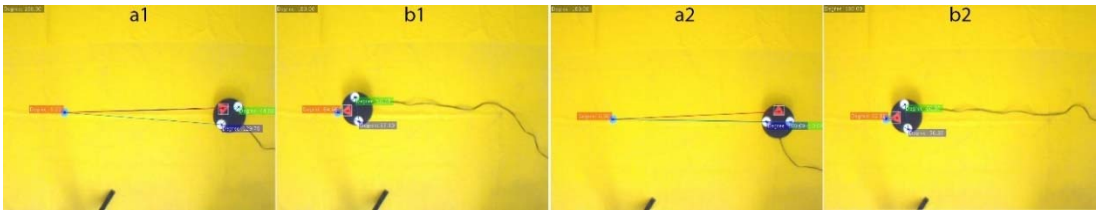

Fig. 48. (a1-a2) Starting and (b1-b2) Finishing position of mobile robot

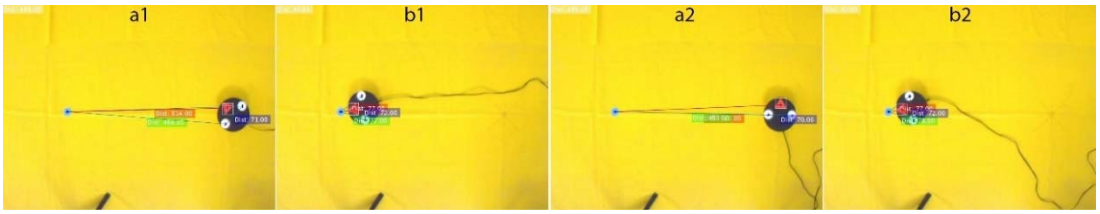

Fig. 49. (a1-a2) Starting and (b1-b2) Finishing position of mobile robot

5.1.5. Comparison of Controller Models

Average experiment results are given in Table 6, Table 7, Table 8 and Table 9. "Total frame" are processed frames from initial to the final position. The "FPS" is frames processed per second. The "Elapsed time" is total time that is emerged due to the control process. The "P. Cost rate" represents the path cost rate that is the difference of percentile rates between the cost of the shortest path and the physical path. The "Energy Cons." represents to rate of consumption for energy. We have performed two non-visual control models to manage the WMR. These commonly known control models are PID and Fuzzy-PID. The default parameters are used in both controllers. It is aimed to see the performance and efficiency issues by comparing both visual and non-visual controllers.

Table 6. '0' degree experiments for controllers

Experiment	Total Frame	FPS*	Elapsed Time**	P. Cost Rate	Energy Cons.
100_0_G	77	8.480	9.08s	0.891%	1.12%
100_0_T	77	7.103	10.84s	0.822%	1.35%
PID	-	-	13.11s	3.283%	2.82%
FUZZY-PID	-	-	11.93s	2.037%	2.51%

Table 7. '180' degree experiments for controllers

Experiment	Total Frame	FPS*	Elapsed Time**	P. Cost Rate	Energy Cons.
100_180_G	86	8.548	10.06s	5.760%	1.47%
100_180_T	92	7.336	12.54s	7.348%	1.84%
PID	-	-	16.34s	10.783%	3.46%
FUZZY-PID	-	-	15.56s	9.545%	3.29%

Table 8. ‘45’ degree experiments for controllers

Experiment	Total Frame	FPS*	Elapsed Time**	P. Cost Rate	Energy Cons.
100_45_G	77	8.521	9.03s	2.708%	1.28%
100_45_T	75	7.217	10.39s	3.324%	1.52%
PID	-	-	14.05s	5.013%	3.16%
FUZZY-PID	-	-	13.44s	4.222%	3.08%

Table 9. ‘90’ degree experiments for controllers

Experiment	Total Frame	FPS*	Elapsed Time**	P. Cost Rate	Energy Cons.
100_90_G	75	8.528	8.79s	3.980%	1.22%
100_90_T	76	7.292	10.42s	5.832%	1.63%
PID	-	-	14.79s	7.425%	3.32%
FUZZY-PID	-	-	14.02s	6.004 %	3.17%

It can be expressed that the graph-based model has better performance than the triangle-based model and non-visual models in aspect of time. It primarily stems from excessive computation processes in triangle-based model. Since calculation of degree values of angles are more complex. Moreover, the triangle-based model has better cost values in the experiments of ‘100_0_X°’ compare to others. However, situation in the experiments of ‘100_180_X’ is exact opposite; graph-based model minimizes the cost more feasibly. The major reason is the simple structure of the graph model. The path cost rate of proposed method is discovered less than the conventional methods. In terms of the energy efficiency proposed method is better than other methods. It should be emphasized that the designed methods are superior to the conventional methods and in terms of efficiency and cost.

Table 10 shows the frame loss in experiments. Main reason of the loss is high level illumination changes. Average frame loss rate has occurred as 1.26%. By designing more efficient image processing methods detection, the performance rate can be improved to the better levels. For example; machine learning methods like ELM, deep learning and other artificial intelligence methods ensure better recognition process compare to primitive methods like shape or color-based detection approaches.

Table 10. Frame loss rates in control process

Experiment	Average Frame	Average FPS*	Frame Loss
100_0_G/T	77	7.791	1 – (1.30%)
100_180_G/T	89	7.942	2 – (2.25%)
100_45_G/T	76	7.869	1 – (1.32%)
100_90_G/T	75.5	7.910	0 – (0.00%)
Average	79.375	7.878	1 – (1.26%)

5.2. STAGE-2: Path Planning Experiments

Experiments have been conducted as both simulation and real-world implementation. Firstly, a configuration space map with position parameters are acquired by processing image of real working environment. Then A-APF simulation has been performed this obtained map and a path plan is extracted. Acquired plan is used to perform WMR motion with decision tree controller in the path trajectory. There are three different configuration spaces in experiments. For each configuration, initial movement direction of WMR is set to '0^o' and '180^o' relative to the position of target. These both directions are selected to see the effect of simple and extreme position conditions. The WMR is located to 110 cm away from the target.

5.2.1. Configuration-1

A. Experiment Conf-1_0

WMR has been positioned towards to the target position approximately with 0^o value. Obstacles have been distributed randomly. Simulation result is demonstrated in Fig. 50. The robot has reached to the target successfully and it takes about 5.692s. Extracted path seems to be safe and efficient. There are several little fluctuations in path, but in fact, they have no remarkable effect on cost. Distance data and potential force changes have been shown in Fig. 51. When distance data shows balanced changes, potential forces have also settled down. Real experiment has taken about 19.762s and it has been demonstrated in Fig. 52. Angle and velocity changes have been given in Fig. 53. Because of velocity is affected by angle values, output of the velocity has demonstrated similar pattern to the angle changes.

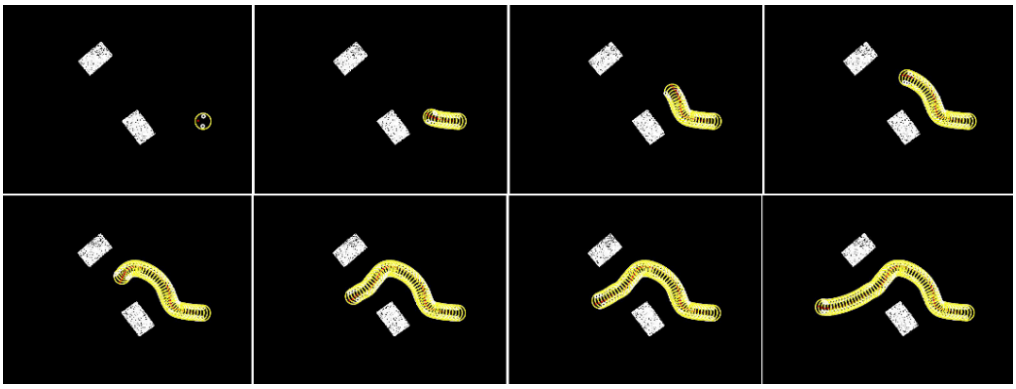

Fig. 50. Simulation result of experiment Conf-1_0

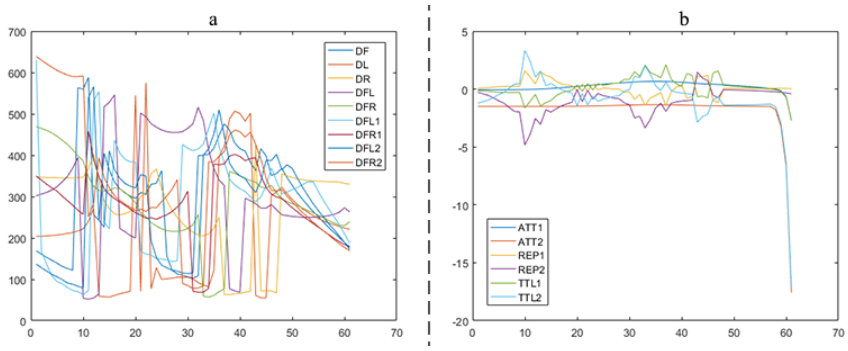

Fig. 51. (a) Sensor data vs. (b) Potential forces

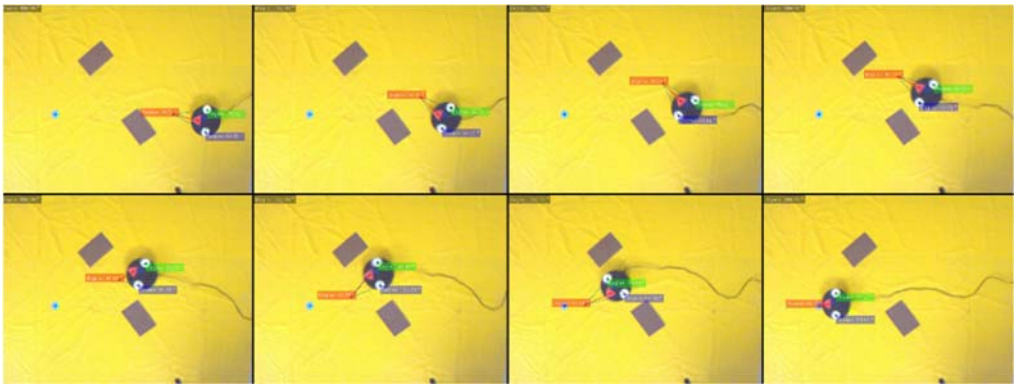

Fig. 52. Real implementation of experiment Conf-1_0

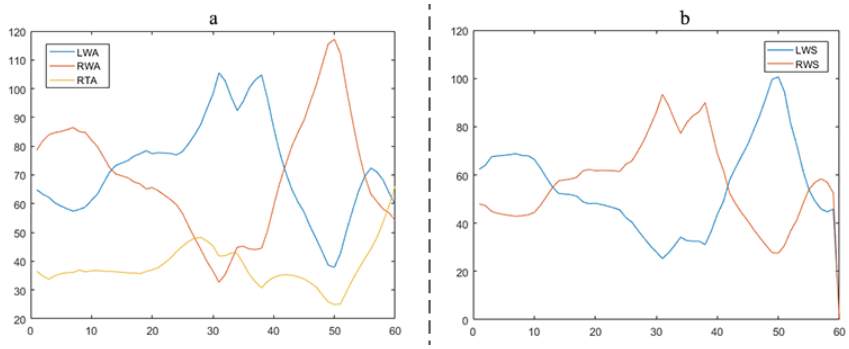

Fig. 53. (a) Angle change vs. (b) Velocity change

B. Experiment Conf-1_180

WMR has been positioned contrary to the target position approximately with 180^o value by previous configuration. The simulation result is demonstrated in Fig. 54. The robot has successfully reached to the target position and it takes about 6.748s. The path has been created safely and efficiently. The acquired distance data and potential force changes have been shown in Fig. 55. As the robot approaches to the obstacle, rapid changes have occurred in potential forces. Real experiment takes 23.799s and it has been demonstrated in Fig. 56. Angle and velocity changes have been given in Fig.

57. Because of velocity is affected by angle values, output of the velocity has demonstrated similar pattern to the angle changes.

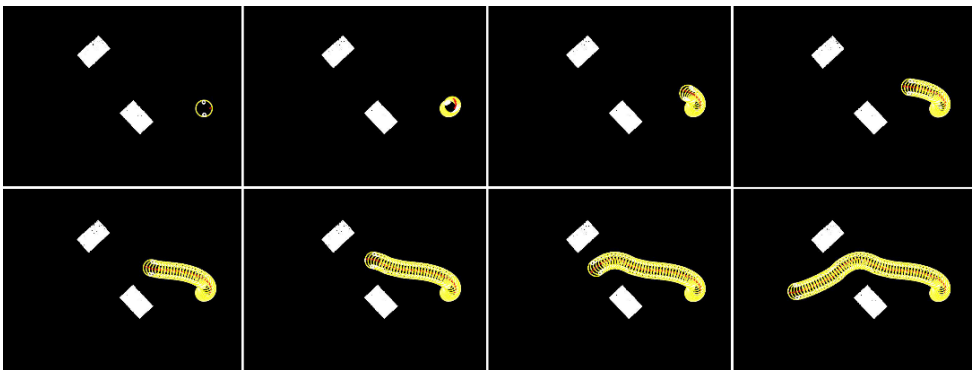

Fig. 54. Simulation result of experiment Conf-1_180

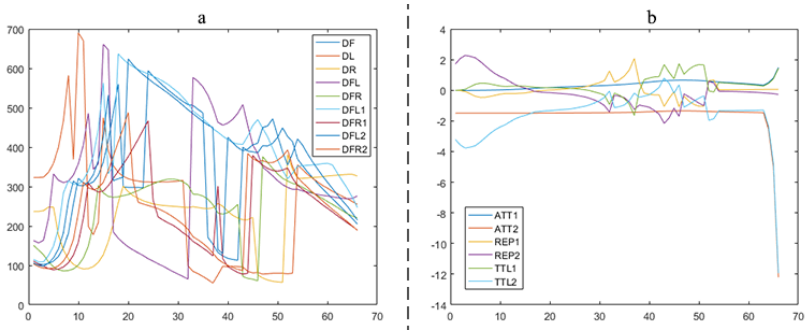

Fig. 55. (a) Sensor data vs. (b) Potential forces

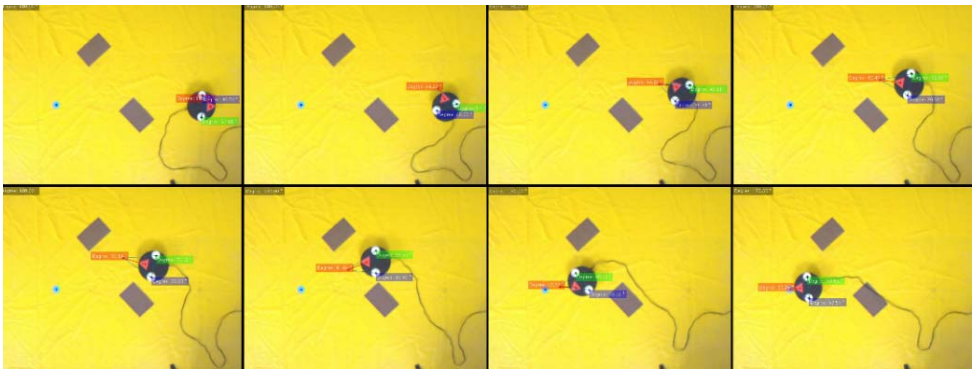

Fig. 56. Real implementation of experiment Conf-1_180

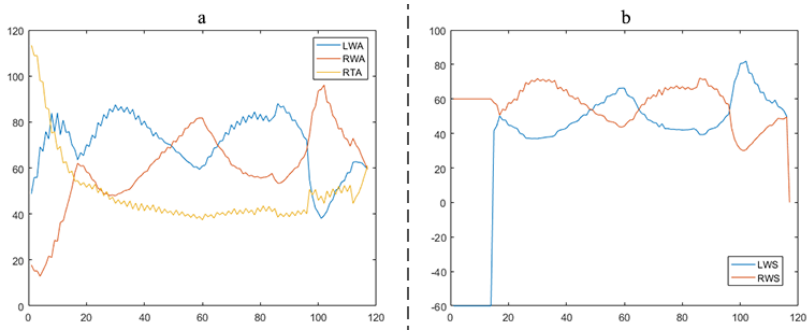

Fig. 57. (a) Angle change vs. (b) Velocity change

5.2.2. Configuration-2

A. Experiment Conf-2_0

WMR has been positioned towards to the target position approximately 0^o with a single obstacle configuration. Simulation result is demonstrated in Fig. 58. The robot has reached to the target and it takes about 6.508s. The path has been formed safely and efficiently. The distance data and potential force changes have been shown in Fig. 59. Potential field have created major forces when moving object approximate to an obstacle. After 24^{th} frame potential forces have exhibited more stable patterns. Real implementation takes about 26.104s and it has been demonstrated in Fig. 60. Angle and velocity changes have been shown in Fig. 61. There is an unexpected angle output and velocity response in 44^{th} frame. This momentary change stems from a miscalculated centroid due to high amplitude of light variation.

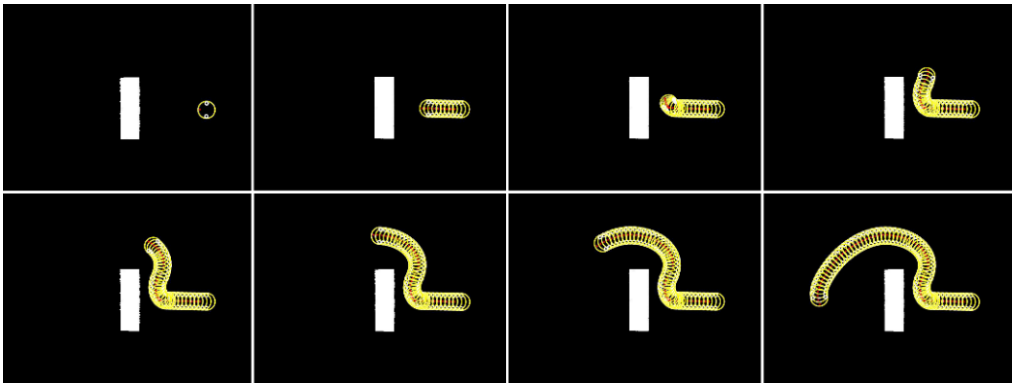

Fig. 58. Simulation result of experiment Conf-2_0

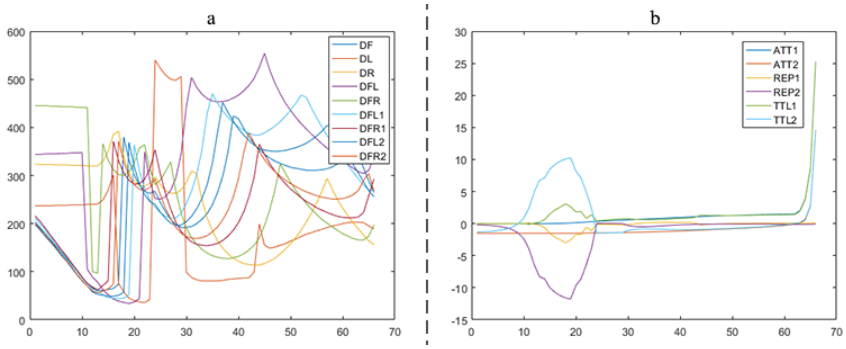

Fig. 59. (a) Sensor data vs. (b) Potential forces

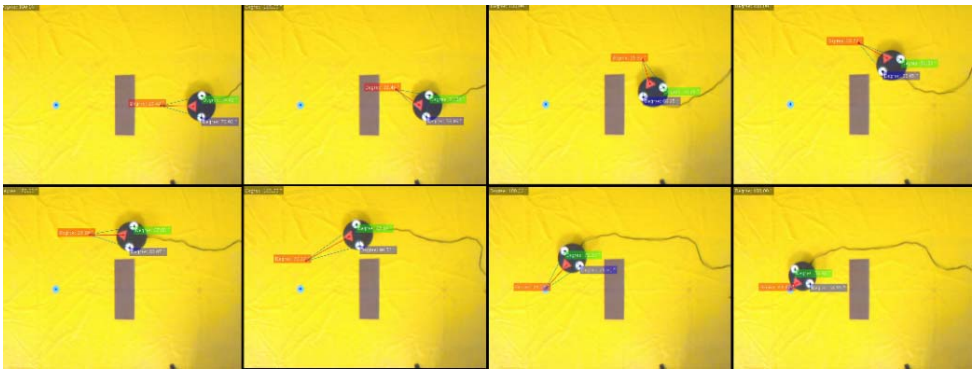

Fig. 60. Real implementation of experiment Conf-2_0

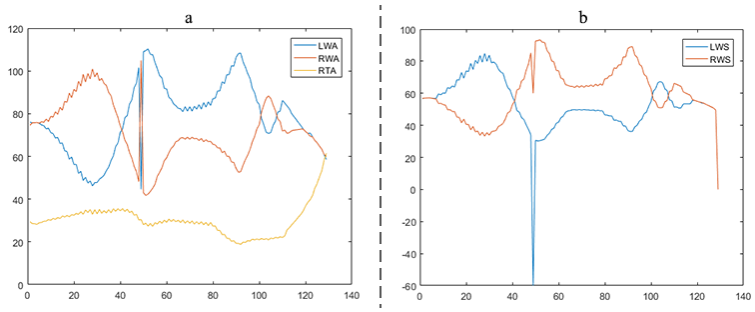

Fig. 61. (a) Angle change vs. (b) Velocity change

B. Experiment Conf-2_180

WMR has been positioned contrary to the target position approximately by 180^o with previous configuration. Simulation outcome is demonstrated in Fig. 62. The robot has reached to the target position and it takes about 7.725s. The path has been emerged as safe and efficient. The distance and potential force changes have been given in Fig. 63. Potential field have generated large forces when the moving object approximate to an obstacle. Stable patterns have been achieved by starting 42nd frame for potential forces. Real implementation takes about 29.520s and it has been given in Fig. 64. Angle and velocity changes have been shown in Fig. 65. As implied before, output pattern of velocity is similar to the output of angle variations.

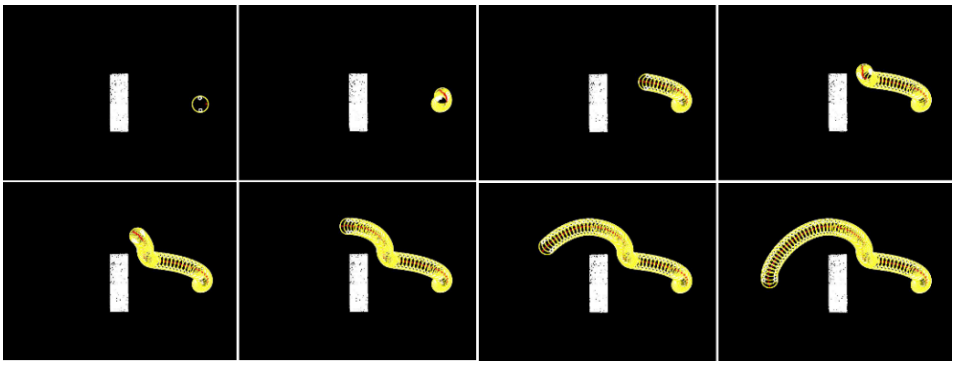

Fig. 62. Simulation result of experiment Conf-2_180

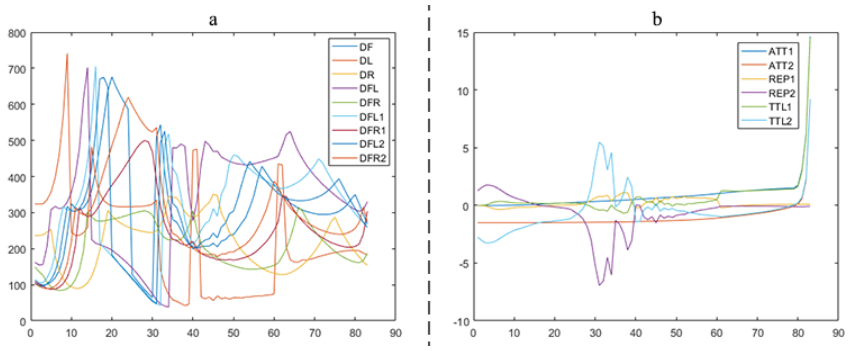

Fig. 63. (a) Sensor data vs. (b) Potential forces

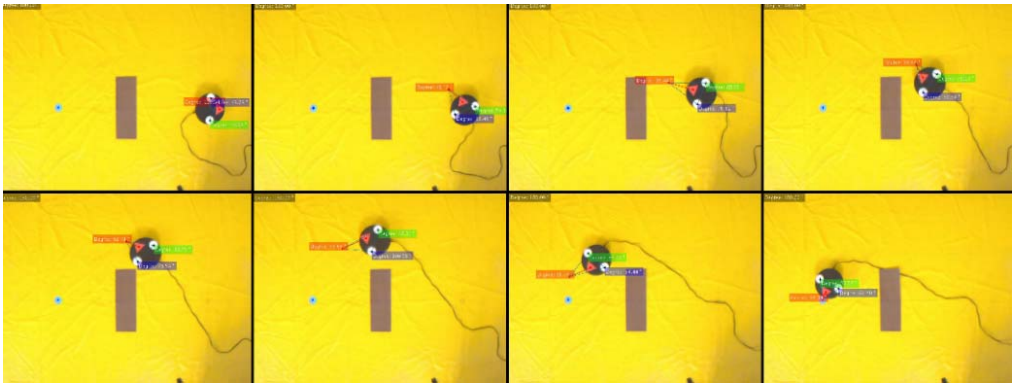

Fig. 64. Real implementation of experiment Conf-2_180

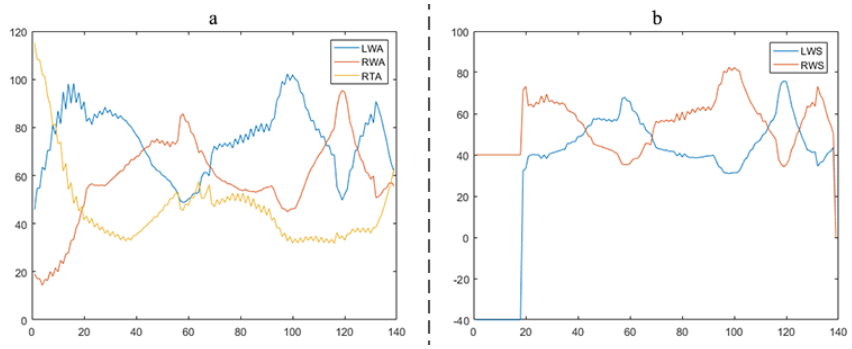

Fig. 65. (a) Angle change vs. (b) Velocity change

5.2.3. Configuration-3

A. Experiment Conf-3_0

WMR has been positioned towards to the target position approximately with 0^o value. The 'U' shaped obstacle has been placed to space. After performing obstacle detection, convex-hull method is used to transform concave object to convex object. The simulation result is demonstrated in Fig. 66. The robot has arrived to the target successfully and it takes about 8.283s. The extracted path is safe and efficient. The distance data and potential force changes have been shown in Fig. 67. Real experiment takes about 32.664s and it has been demonstrated in Fig. 68. Although

simulated path is safe, there is a risky approximation to the obstacle in real implementation. The angle and velocity changes have been given in Fig. 69.

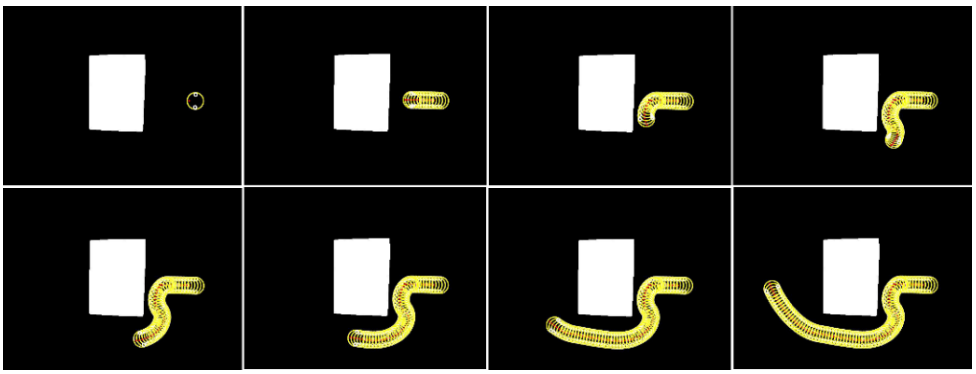

Fig. 66. Simulation result of experiment Conf-3_0

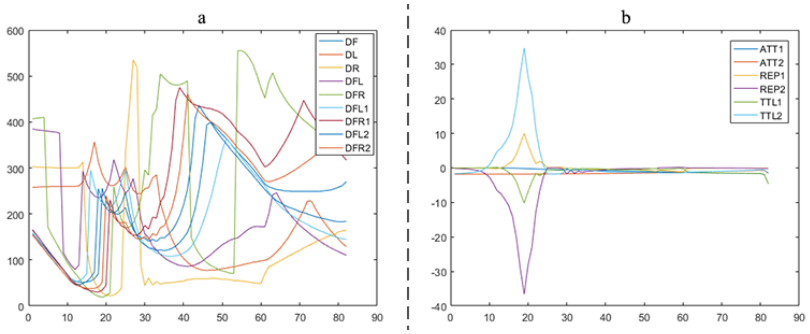

Fig. 67. (a) Sensor data vs. (b) Potential forces

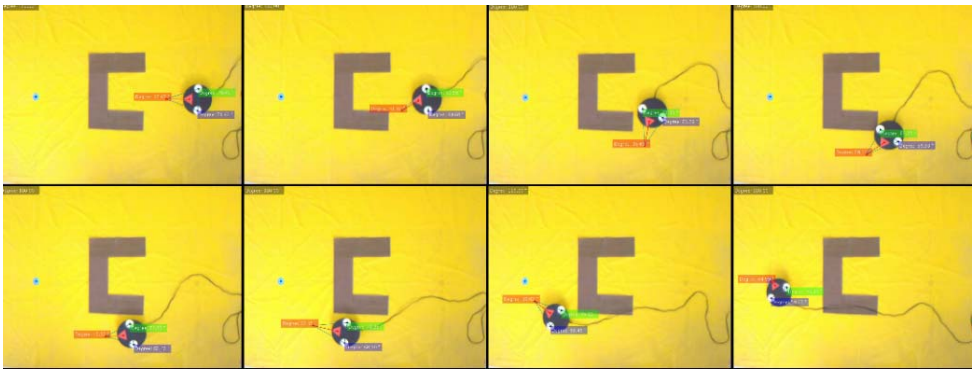

Fig. 68. Real implementation of experiment Conf-3_0

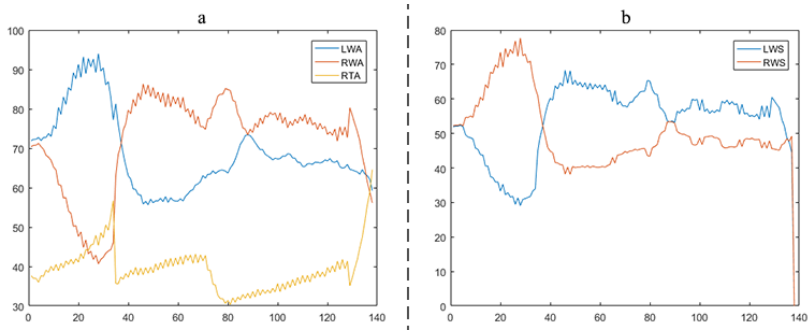

Fig. 69. (a) Angle change vs. (b) Velocity change

B. Experiment Conf-3_180

WMR has been positioned contrary to the target position approximately by 180^o value with previous configuration. Similarly, convex-hull method is used to convert concave object to convex object with minimum edge boundaries. Simulation result is demonstrated in Fig. 70. The robot has arrived to the target position successfully and it takes about 9.566s. The path is extracted safely and efficiently. The distance data and potential force changes have been given in Fig. 71. Real implementation takes about 34.895s and it has been shown in Fig. 72. The previously mentioned risky approach to the obstacle is eliminated by using object dilation. By this way path safety is increased. Angle and velocity changes have been given in Fig. 73.

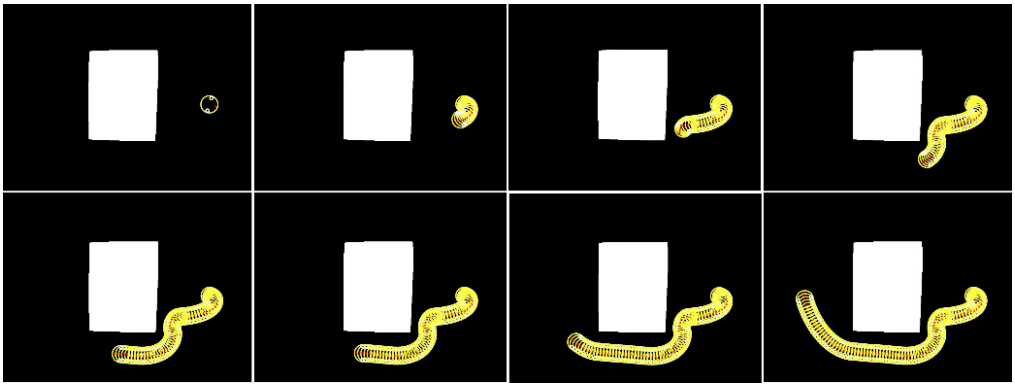

Fig. 70. Simulation result of experiment Conf-3_180

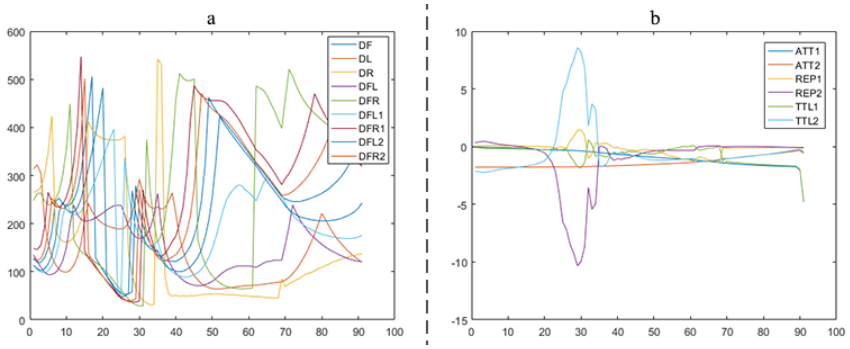

Fig. 71. (a) Sensor data vs. (b) Potential forces

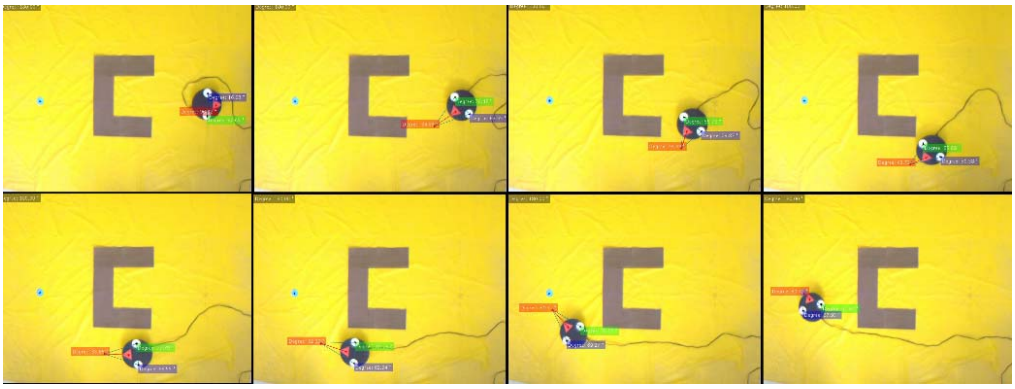

Fig. 72. Real implementation of experiment Conf-3_180

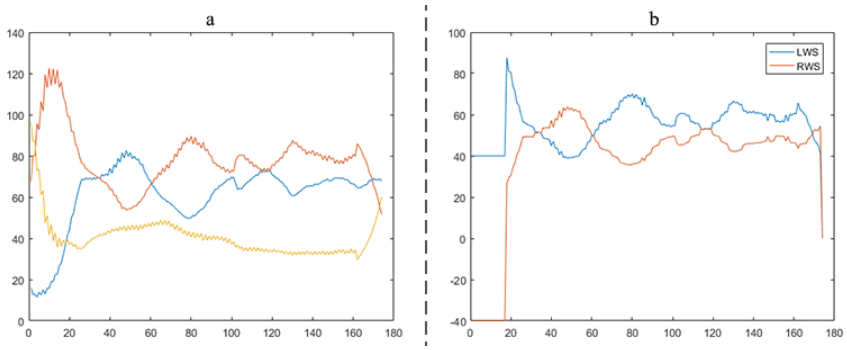

Fig. 73. (a) Angle change vs. (b) Velocity change

5.2.4. Experiment Comparisons

The experiments have been repeated three times for each configuration. Average values of these experiments data are given in Table 11. If the configuration complexity has been increased for potential fields, arriving time to the target has also increased in both simulation and implementation, generally. Similarly, path cost has also increased by depending on complexity of configuration space. When the robot is positioned contrary to the target position, the path cost is expected to increasing. In Conf-1 experiment path cost did not remarkably change. In Conf-2 experiment, path cost is increased with contrary direction. However, in Conf-3 experiment path cost decreased with contrary direction; this is because, sometimes turning around to the target help decreasing the path cost. Since, the WMR did not go toward the middle of obstacle directly. On the other hand, the implementation path cost is always less than simulation path cost for all experiments. Because, the decision tree controller drives WMR to the continually updated positions of middle-targets (or local targets) by minimizing the errors. It smooths sharp turns stemming from APF path planning. In other words, the local solutions give better results than global solutions. By this way, path cost has decreased. The simulation speed is set to larger than implementation

speed. Because we want to extract path plan as fast as possible. Average path cost gain actualized about 11.335%.

Table 11. Simulation and Real Implementation Comparison

Experiment Configuration	Simulation Time	Implementation Time	Simulation Path Cost (px)	Implementation Path Cost (px)
Conf-1_000	5.692s	19.762s	486.268	442.356
Conf-1_180	6.748s	23.799s	485.874	439.145
Conf-2_000	6.508s	26.104s	619.761	548.210
Conf-2_180	7.725s	29.520s	644.236	562.661
Conf-3_000	8.283s	32.664s	753.703	656.506
Conf-3_180	9.566s	34.895s	730.276	640.700

5.2.5. General Observations

The path plan has extracted with a simple adaptive APF. The decision tree-based control has been successfully operated in different configuration spaces. The control method does not produce any remarkable bottleneck to entire control process. The robot has accurately arrived to the target position for each configuration. As the robot approaches to an obstacle, distance values measured around the robot are decreased, so the potential forces give sensitive responses. Potential forces have stabilized as WMR move away from the obstacle. Eventually, path plan is extracted with A-APF method successfully. Method success has been investigated under different circumstances like obstacle positions, variable ambient light. After gaining a suitable path from initial position to the target position, this path has been given to the real-time control process as trajectory input. Decision tree-based controller successfully manages the WMR with acquired obstacle-free path plan in real-time. Angle values have been correctly calculated. Therefore, velocity have been acquired smoothly in each control iteration.

5.3. STAGE- 3: Multi-Camera Experiments

5.3.1. Experiment Configurations

Multi-camera experiment has been performed with four webcam cameras. In single camera tests, a single CCD camera is used as emphasized before. The CCD camera provides more color depth compare to webcam, but it is very expensive. Therefore, this cost is needed to be reduced for a multi-camera configuration. On the other hand, webcams are easy to use, platform independent in terms of software and hardware,

adaptable to any environment with proper configuration. Of course, the CCD cameras have their specific advantages. However, such properties do not be required for this WMR control configuration.

Each webcam is adjusted to SVGA resolution. Cameras have been hanged to the ceiling in a way that their lenses have been located as perpendicular to the floor with aluminum support profiles. Cameras have been placed about 210cm from the floor. According to this camera position each camera covers about 3.05 m^2, so in total approximately 12.20 m2 area should be processed for this configuration. It should be noted that all the webcams are same models and they have same specifications. Each camera is connected to the computer through USB 2.0 ports, separately. This physical configuration has been demonstrated in Fig. 74.

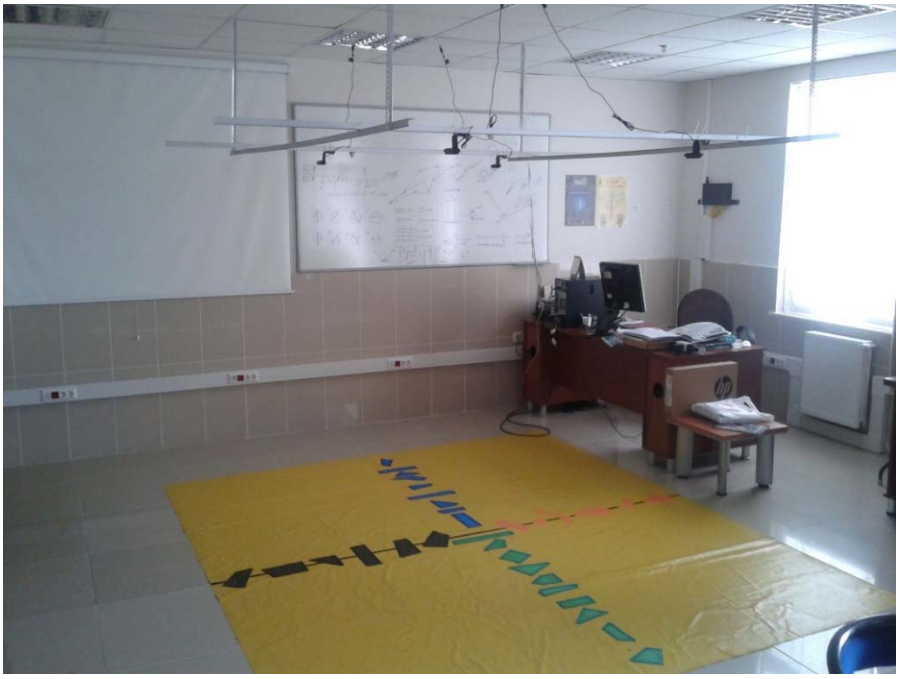

Fig. 74. Real multi-camera based WMR control operating environment

Several different colored and shaped labels have been placed to the floor as distinctive properties for SURF detector. Because the SURF detector searches and locates similar properties according to given input image. The size and color of shapes have been randomly determined. The only important factor is position of these labels. Each camera covers a single plain area including two axes labels according to camera position as shown in Fig. 75.

Fig. 75. Colored and randomly shaped labels on the operating floor

The webcams (Logitech C920) used for multi-camera configuration is shown in Fig. 76. They are attached to aluminum profiles with plastic clips. It has 3.2MP maximum video resolution and 15MP image resolution. However, SVGA (800x600) is used for visual servoing task. Therefore, even more basic webcams will be enough to deal with the indicated resolution.

Fig. 76. The webcam used to perform multi-camera configuration

Camera positions have been shown in Fig. 77. IDs are given as C1, C2, C3 and C4 to the cameras respectively. Camera viewing area is indicated with blurred area for C1. C1-X and C1-Y are the length and height of rectangular C1 camera viewing area (CVA). Other cameras have similar viewing areas according to their positions. The blue and red areas represent the common intersection areas for two webcams. The middle square area represents intersection area of four webcams. Black lines represent guidelines.

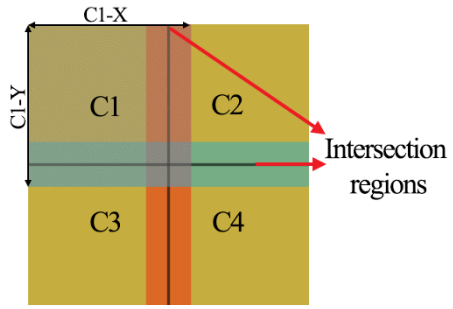

Fig. 77. Camera positions and camera intersection areas

5.3.2. Multi-Camera Experiment with Conf-1

Images taken from the cameras from bird's eye view configuration are shown in Fig. 78. The grayish areas on the left or right edges in the images are real floor texture of the experiment environment. The utilized mobile robot steering wheels are quite thin and ball caster wheels are small. Therefore, plastic based yellow layer is used to prevent wheel jamming to suture area of the floor tiles.

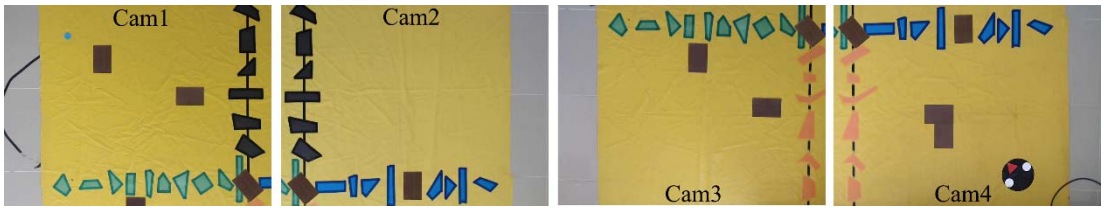

Fig. 78. Real areas covered and acquired by the cameras

The stitched images are demonstrated in Fig. 79 as the first configuration (Conf-1). Images are simply superimposed and re-scaled. To increase accuracy of SURF detector, colored labels are used. Eventually, images are stitched to each other successfully. The brown colored objects represent the obstacles. Robot is under the C_4 camera CVA and the target is placed under the C_1. It can be seen that there is different level of shadowing in the images taken from the cameras. Because of such differences, several negligible inconsistencies in stitched objects have emerged. They

are negligible; since, all these errors are too small to be effective on path planning and visual servoing tasks.

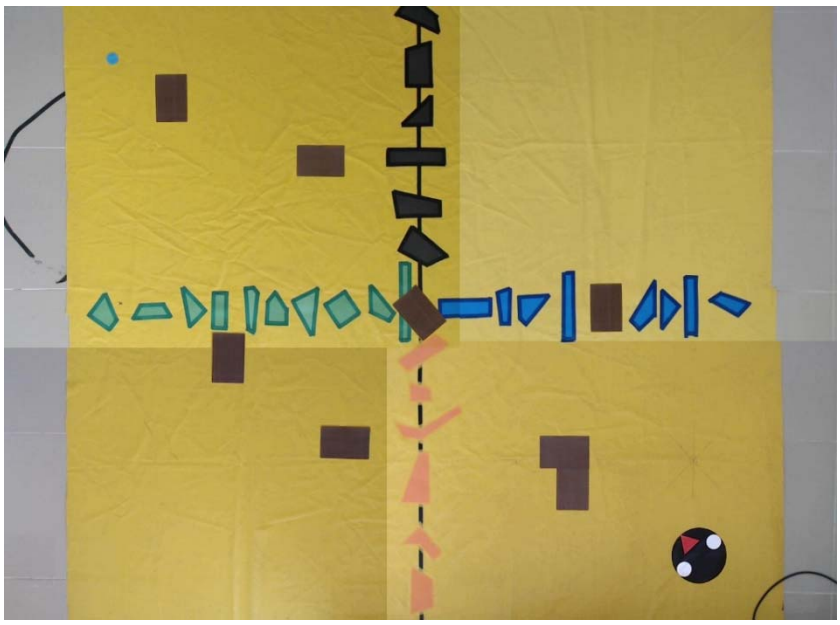

Fig. 79. The stitched image to acquire Configuration-1 (Conf-1)

After acquiring stitched environment, obstacle detection task is executed as in single camera configuration, Fig. 80. The obstacles are detected and the environment is converted to binary map. This process is performed by assigning '1' to the obstacles and assigning '0' to the remaining area. This task is known as 'Binary Image Acquisition'. The robot and target positions are also detected and stored. To increase safety, the object dilation is used to re-scale detected obstacles.

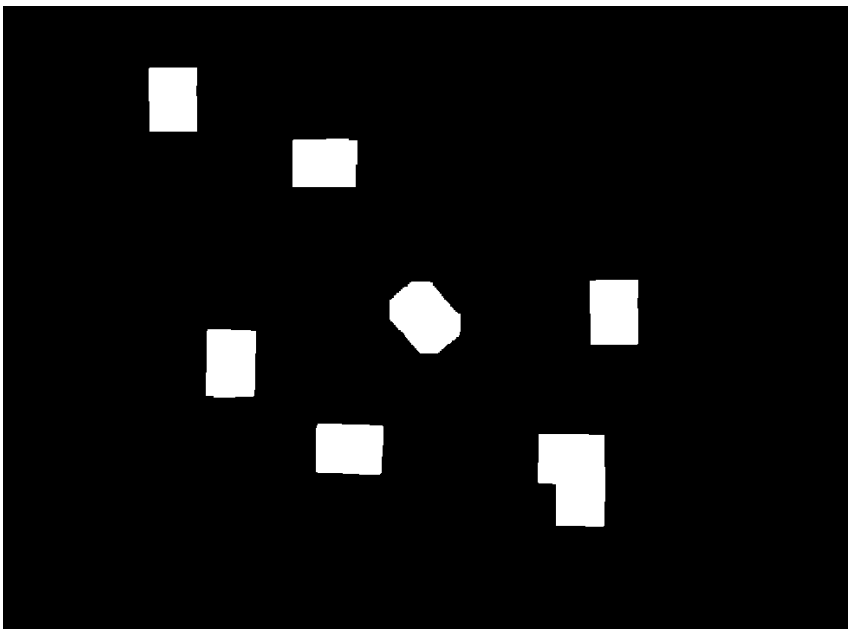

Fig. 80. Obstacle map acquired from the stitched image

Adaptive artificial potential field (A-APF) method is performed the path planning process on acquired map. The extracted path has crossed on three cameras. The $P_{tq} = (C_4, C_2, C_1)$ and 192 image frames have been processed. Therefore, 192 different position sampling has been taken on the acquired path. These positions are used to implement visual based control process with designed controllers. Simulation takes about 11.2s, so 17.142 frames per second is obtained. The simulation path cost is found as 1037.53px. The Gaussian controller with triangle positioning scheme is manage the robot to approach to the target position, successfully. The next suitable position is calculated in each iteration. In Fig. 81, the formed path by A-APF is given.

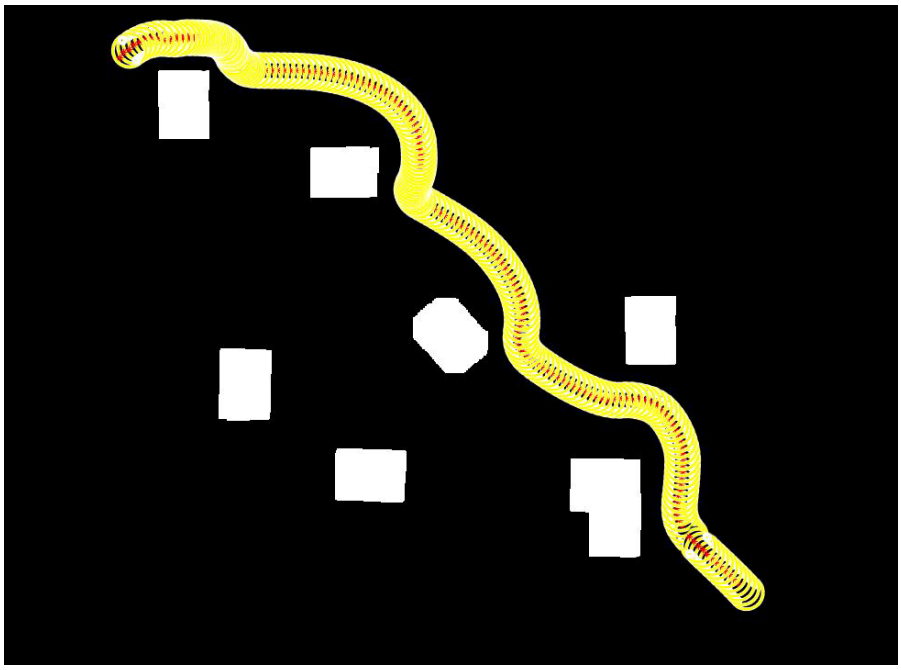

Fig. 81. Simulation path with A-APF

Attractive potential field (A-PF), repulsive potential field (R-PF) and total potential field (T-PF) force values against number of processed image frames are given in Fig. 82. A-PF force increases at several frames from the starting, then it decreases until the target position is reached. On the other hand, R-PF forces show changing pattern until the assigned task is completed. T-PF forces is formed by combining attractive and repulsive forces. As it can be seen, total forces are quite similar to the opposite direction values of the A-PF forces.

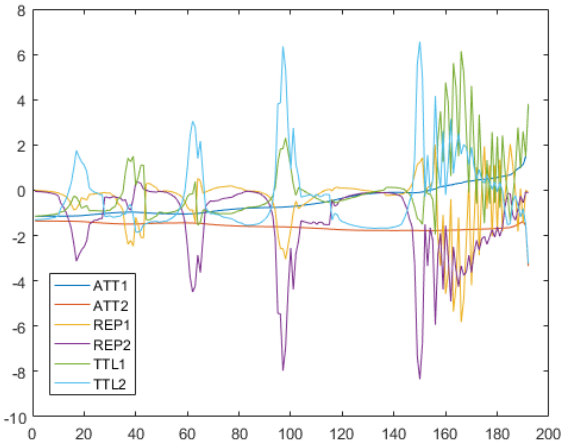

Fig. 82. Potential force change

The changes of attractive and repulsive gain values (aps = ζ and rps = η) are given in Fig. 83. The 'aps' increases for a while from the start point, then it decreases with small rates as iteration continues. The 'rps' increases aggressively at first, then it decreases almost vertically to a point. It approaches near stabilize state with a little fluctuation until the end of the simulation. On the other hand, potential calculating order shows small changes and minimum calculating order shows no-changes.

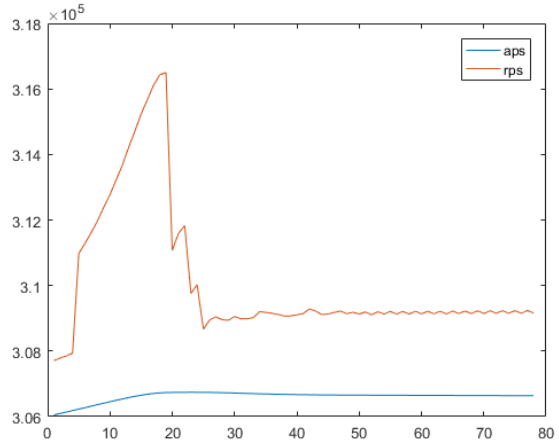

Fig. 83. Potential scaling factors change

Real implementation frames under C_4 are given in Fig. 84. The 'f1, f2 … f8' frames show different robot positions at different times. In Fig. 85, the C2-s, C2-f and C1s, C1-f represent the starting and final position under C_2 and C_1, respectively. The simulated and real paths are given in Fig. 85 as well. The 153 frames are processed in total (with all sub-paths in P_{tq}). Moreover, 14.57 FPS is achieved with 10.5s time for Conf-1. Only one-third of the total frames are stored to keep performance stable.

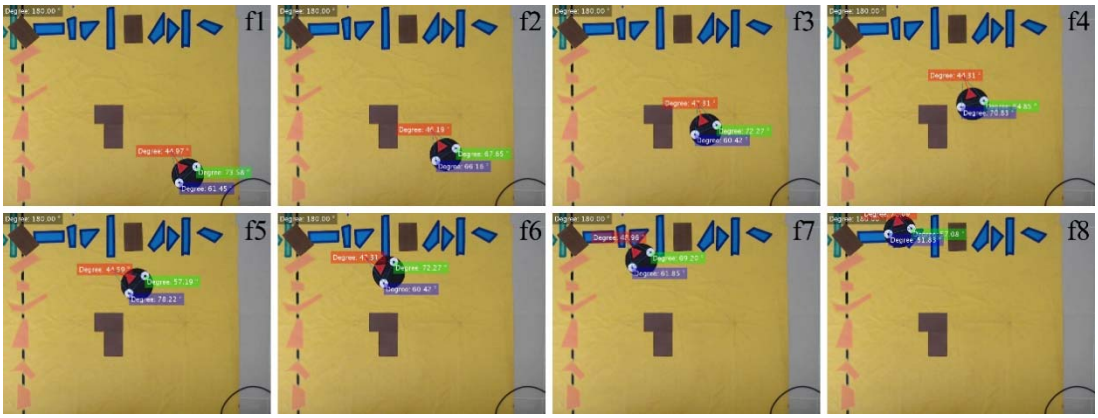

Fig. 84. Sample frames from visual based control task under C_4 camera

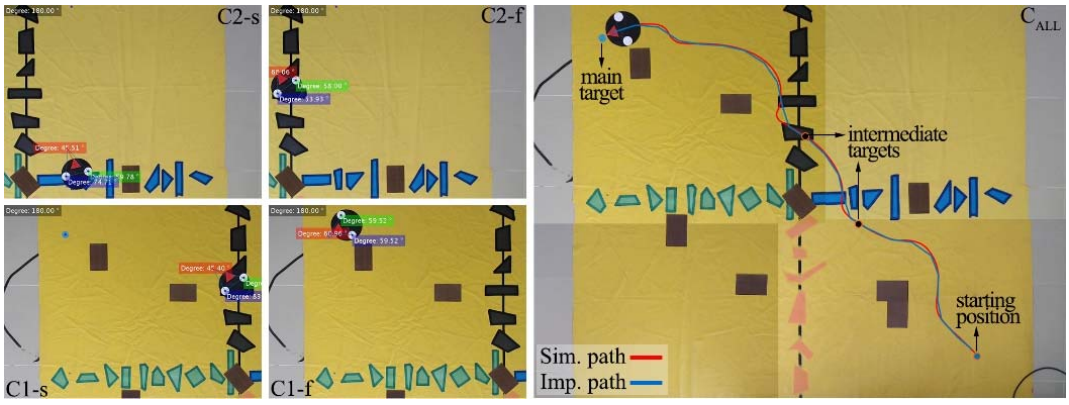

Fig. 85. (I) Robot positions under C2 and C1 cameras (II) Simulation and Real paths

Acquired path plan has been used as reference path which have to be followed by the mobile robot. The robot is triggered to make motions according to reference path in real time. A_R, A_L and A_T values are calculated as 73.81º, 69.19º and 37.0º respectively according to the intermediate target at the first starting frame. These values are calculated as 59.29º, 61.41º and 59.29º respectively at the end of the control task. Robot has successfully reached to the pre-defined target about 10.5s. Starting and finishing positions of the mobile robot is given in Fig. 86.

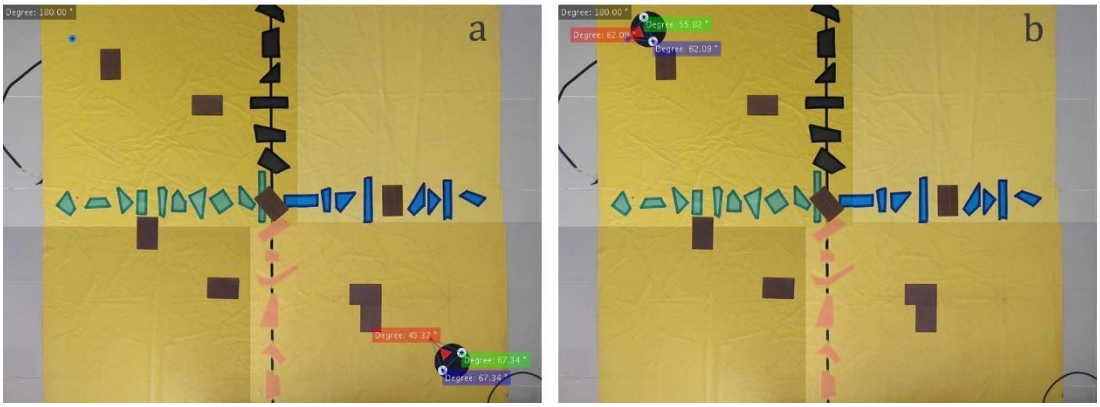

Fig. 86. (a) Starting position and (b) finishing position of the mobile robot

Sample frames from the visual control process in the whole working environment are given in Fig. 87. The ‘f1, f2 … f8’ frames demonstrate different robot positions at different times.

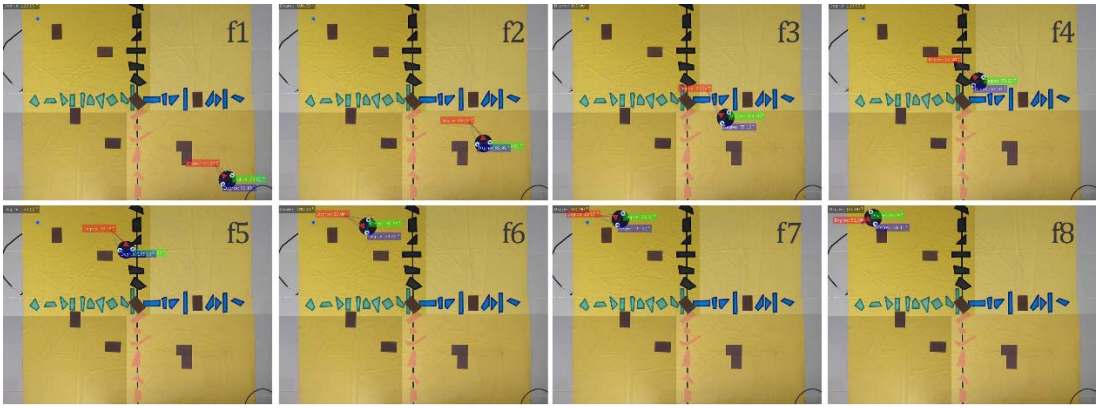

Fig. 87. Sample frames from visual based control task

The path created by robot motions are given in following Fig. 88. The controller has tried to kept the mobile robot on acquired path through the control process. The distance of path created by robot motions until to the target position is emerged a little smaller than the distance of simulation path. Main reason behind this situation is the dynamically changed local targets used to track the simulation path. Local target is extracted from simulation path within a pre-defined threshold value and it is periodically updated until reaching to the main/final target. In this way, the controller generally smooths sharp turns.

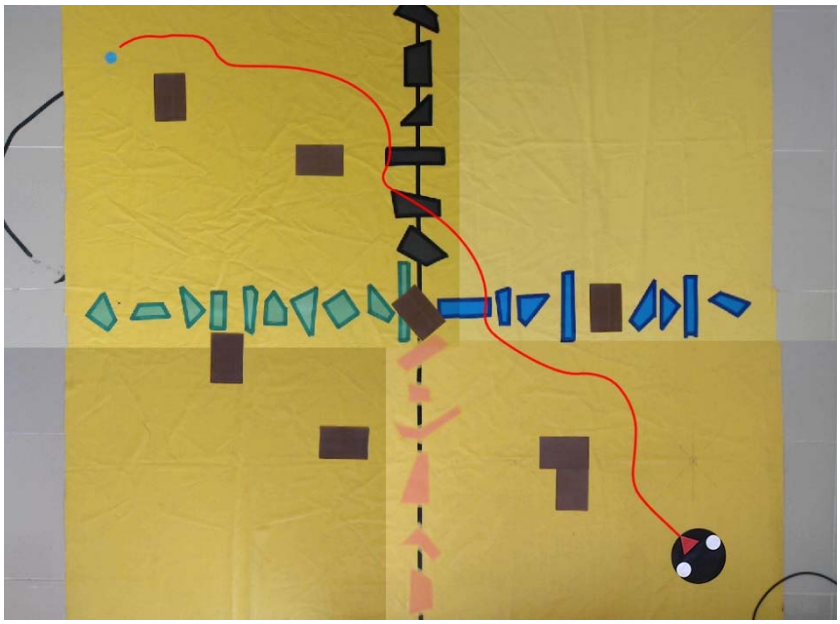

Fig. 88. Simulated path and starting position of robot

The path and robot motions from selected frames are demonstrated in Fig. 89. Except from starting and finishing positions of robot, several additional intermediate positions have been given. Eventually, the mobile robot has smoothly tracked the input path. There may be some error between simulation and real path. However, this error is so small in terms of path cost, so it is negligible.

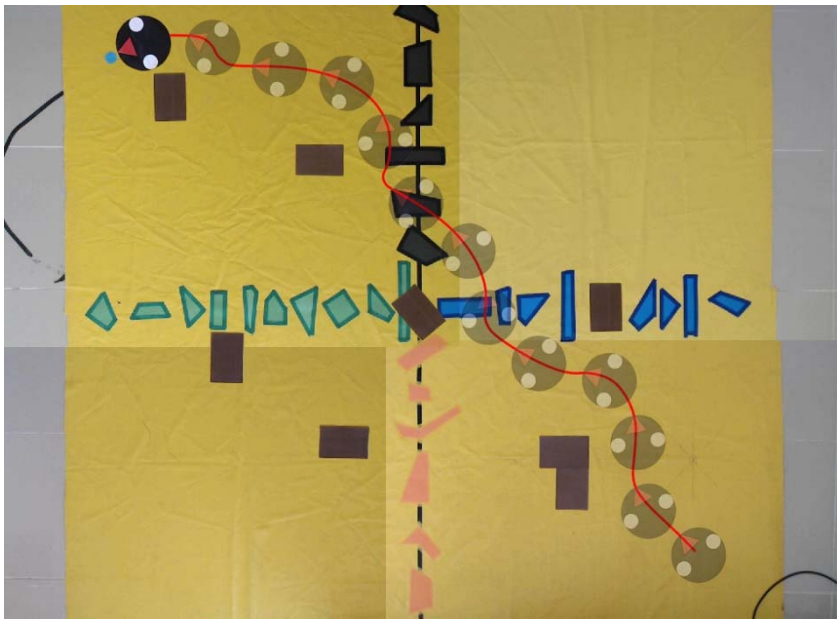

Fig. 89. Simulation path and mobile robot motions

The real path formed by the mobile robot has been given in Fig. 90. As it seen, the distance of real path is emerged a little smaller than the simulation path. Its length is found about 995.16px. Therefore, there is only 4% difference between paths.

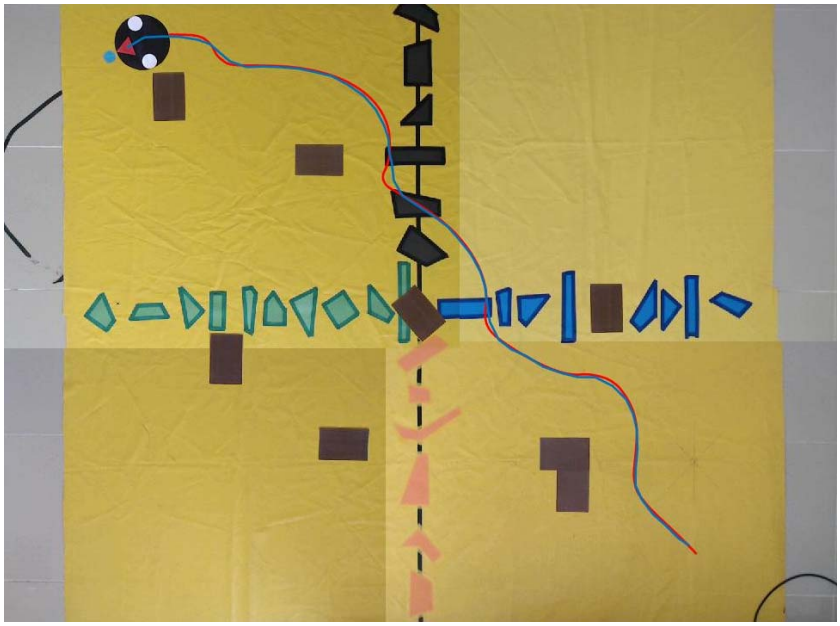

Fig. 90. Simulation path (red) and Real path (blue)

The angle value changes of the control points (mobile robot wheels) and target are given in Fig. 91. The local target point is controlled in each iteration and if it is required, this target position is updated. The angle changes have dramatically increased when the mobile robot starts to perform turning motions. At the end of the control process A_T, A_L and A_R angle values approach to the each other very closely. This means that the robot gradually approaches to the target position.

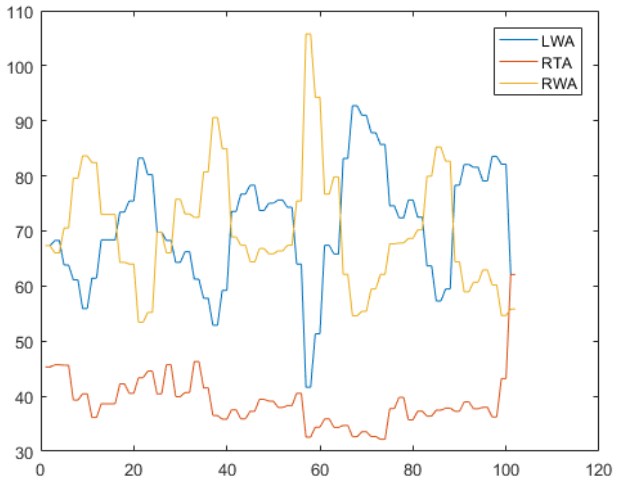

Fig. 91. Angle changes of control points

Velocity changes of the left and right wheels are given in Fig. 92. The changes in velocity values look like the changes in A_L and A_R angle values with different magnitude. The main reason is that the angle values directly affect the velocity values of mobile robot wheels. Both the angle and velocity value changes are a bit jagged. This is because; sensitivity of the controller and storing of selected sample frames to the disk.

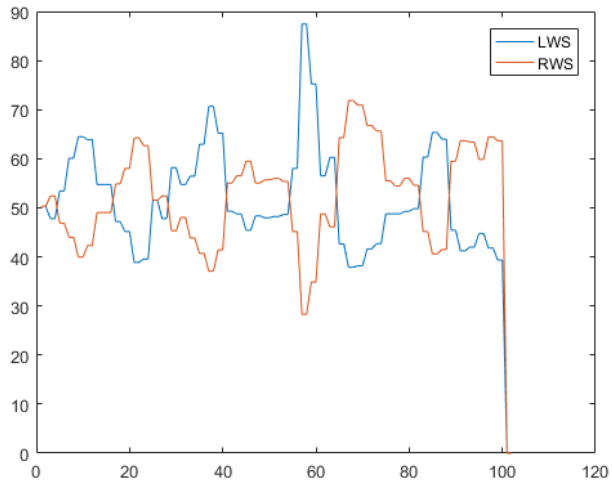

Fig. 92. Left and Right velocity changes of WMR wheels

5.3.3. Additional Experiments with Different Configurations (Conf-2/3)

Experiments have been performed on two different operating environment configurations beside the previous configuration (Conf-1). The experiment environment and acquired path plan with A-APF method is given in Fig. 93 for Conf-2 and Fig. 94 for Conf-3. In each configuration, the object dilation to the obstacles has been implemented to increase path safety. Only the acquired paths and numerical results have been given in these experiments.

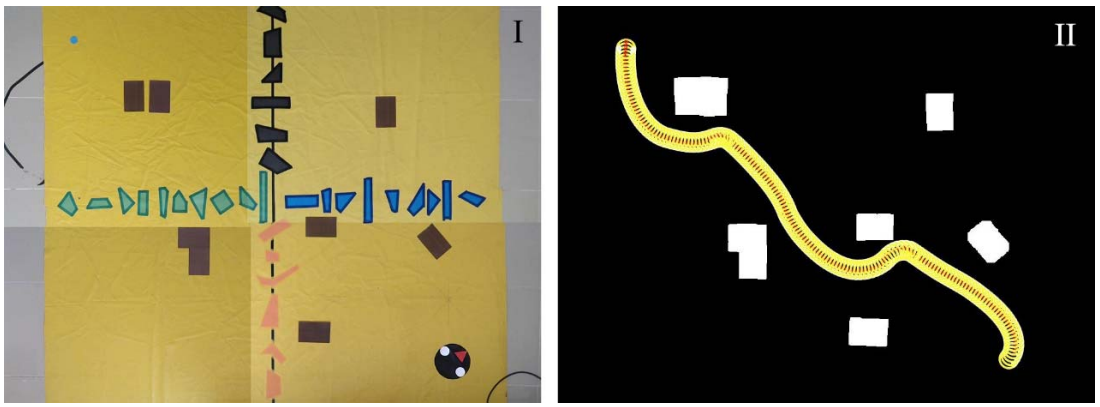

Fig. 93. (I) Configuration-2 (Conf-2) and (II) simulated path plan

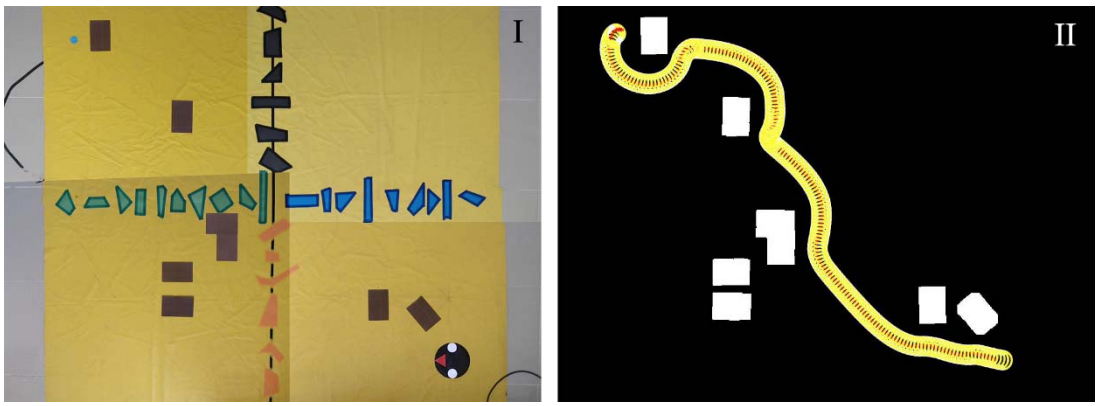

Fig. 94. (I) Configuration-3 (Conf-3) and (II) simulated path plan

The mobile robot has successfully reached to the pre-defined target under configurations with different obstacle alignments in simulation experiments. In each configuration images taken from camera can be superimposed differently. The important principle is the fusing common intersection areas with a high degree of precision. The starting and finishing positions with angle values by the mobile robot are given in Fig. 95 for Conf-2 and Fig. 96 for Conf-3. In Conf-1, the WMR has started with $A_T = 14.94°$, $A_L = 142.30°$ and $A_R = 22.76°$ and reached to the target with $A_T = 57.34°$, $A_L = 61.33°$ and $A_R = 61.33°$ angle values. On the other hand, in

Conf-2, the WMR has started with $A_T = 32.47°$, $A_L = 65.88°$ and $A_R = 81.65°$ and reached to the target with $A_T = 57.03°$, $A_L = 55.80°$ and $A_R = 67.17°$ angle values.

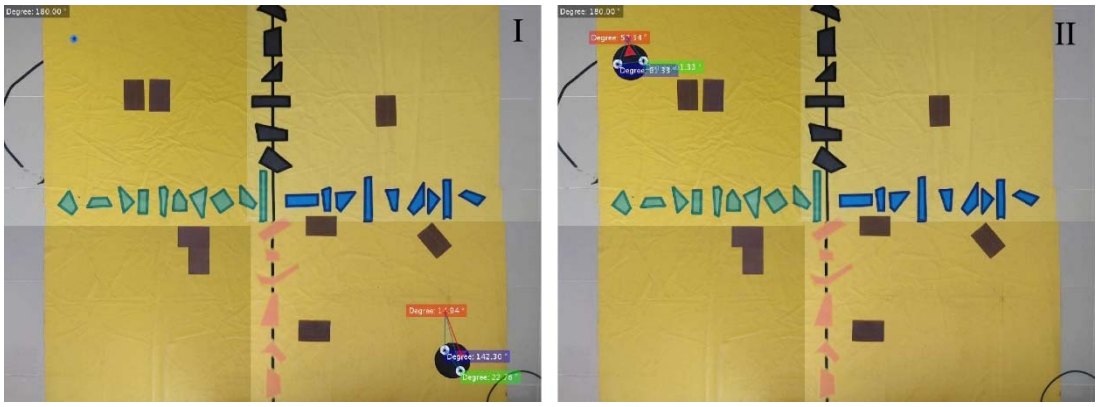

Fig. 95. (I) starting position and (II) finishing position for Conf-2

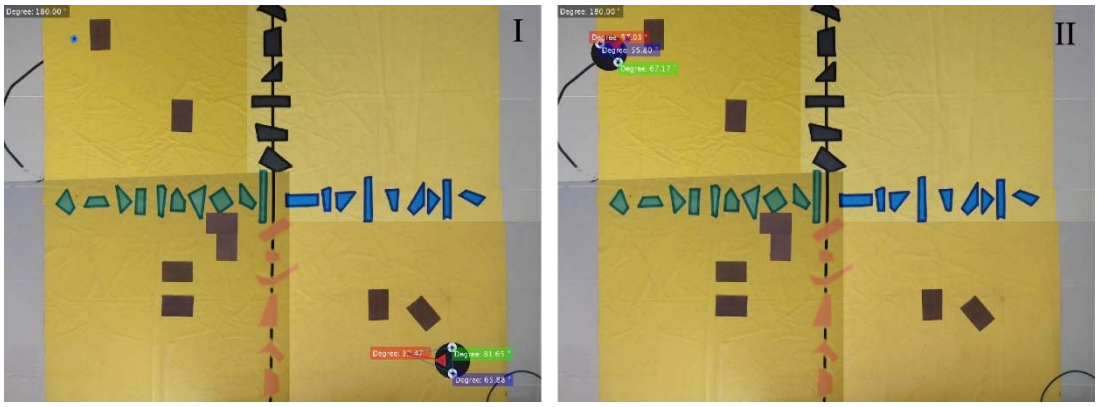

Fig. 96. (I) starting position and (II) finishing position for Conf-3

The formed path by the mobile robot from starting position to finishing position is given in Fig. 97. In each configuration the object dilation to the obstacles has been applied to increase path safety.

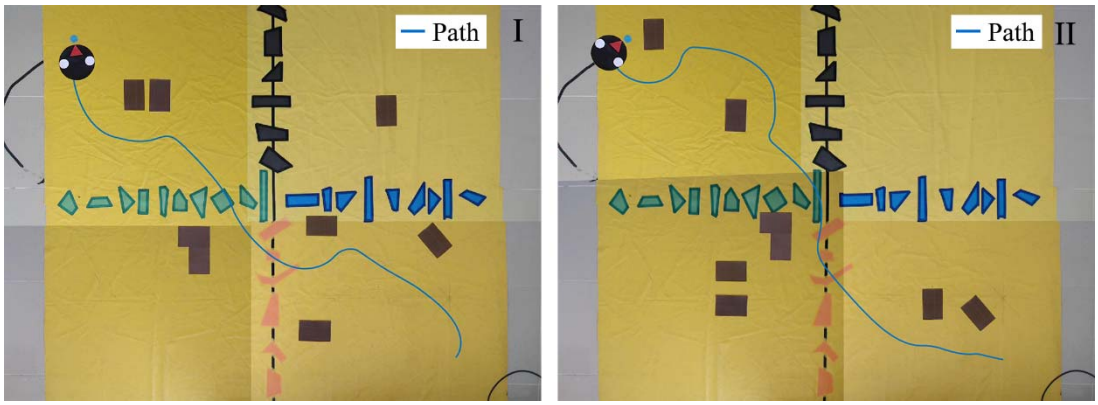

Fig. 97. (I) path formed in Conf-2 (II) path formed in Conf-3

Acquired time and path cost values are given in Table 12. Except the first experiment (Conf-1) simulation is performed faster. The real path cost is generally

smaller than simulation path costs. The average difference between simulation and real path costs has emerged about 3.656% for all experiments. Both simulation and real paths are not the best paths in terms of cost and safety. However, it can be said that acquired paths are close to the best solution and provides feasible balance between the cost and safety.

Table 12. Acquired time and cost values for different configurations

Experiment	Simulation Time (s)	Implementation Time (s)	Simulation Path Cost (px)	Real Path Cost (px)
Conf-1	11.2	10.5	1037.53	995.16
Conf-2	11.4	11.9	1088.65	1055.42
Conf-3	12.5	13.8	1143.08	1099.27

5.3.4. General Observations

Multi-camera configuration has some advantages and disadvantages. Advantage is that the sensor-in-device (or eye-in-device) hardware are not needed. Therefore, cost of the system can be reduced. Second advantage is that working space can be enlarged with additional cameras. Third advantage is that all the robots can be controlled from one system (actually system may be saturated to an upper limit). Disadvantage of the designed system is that it can be only established for indoor environment. The number of required cameras may be high for large interior environments. So, the flexibility of the system will reduce. To overcome this problem, according to applying area, focal length and resolution of cameras and light intensity have to be tested and adjusted.

Image stitching is a time-consuming task, but this task only performed once for every mobile robot control task. The illumination and shadowing can also be the problematic issues for multi-camera configuration. Although all the cameras have same specifications and have been placed to the same height, images taken from these cameras can have different level of shadowing and illumination. Therefore, these two problems should be focused further in eye-out-device robot control systems.

On the other hand, the control process can be done without image stitching task under multi-camera configuration. This process can be done by modeling intersection areas as middle-target points. The mobile robot can progressively reach its final position by using these middle targets as starting/finishing positions.

5.3.5. Experiments without Image Stitching (Conf-1)

Multi-camera images have been stitched in previous experiments. It provides accurate results but it is a time-consuming task. Since, before performing the control process all images are stitched by utilizing SURF detector. Then path is extracted from this whole image. The acquired path positions are distributed according to the camera coverage area where the WMR will appear. For instance, two cameras may be enough to deliver the mobile robot to the desired target position. Acquired images from the cameras are demonstrated in Fig. 98. It is shown that the mobile robot is initially positioned under the C4 camera and main target is fixedly positioned under the C1 camera. The configuration space is the same space used in the first image stitching based multi camera experiment.

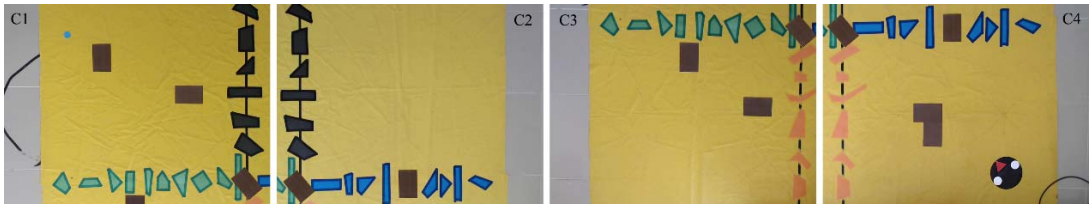

Fig. 98. Real acquired areas covered by the cameras

This time images are not stitched. Each of the camera images are considered local maps that includes local initial and target positions. Local target is determined according to the most suitable intersection area which is closest to the main target and has enough space for WMR. When the WMR reaches to the local target position in first camera coverage area where it resides, this local target point is assigned as initial position for WMR in next camera which closest to the main target position.

The local target determination process is illustrated in the following Fig. 99. The direction information is determined relative to the target and initial positions of the mobile robot. Therefore, it can be said that the main target is in NW (North-West) direction.

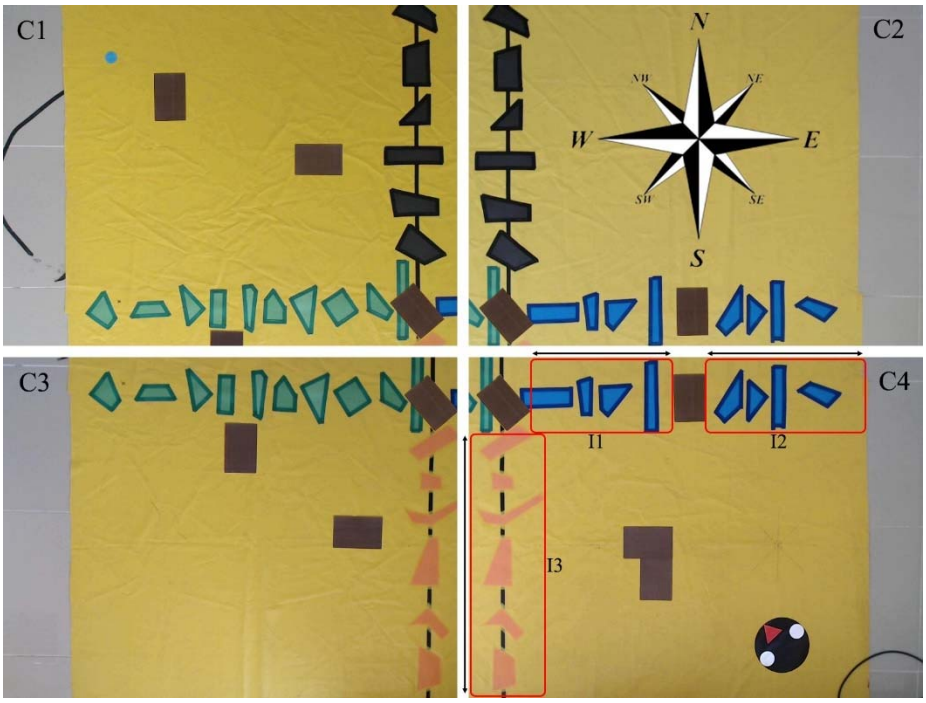

Fig. 99. Obstacle-free intersection regions for a camera (C4)

After determining the directional information, the closest intersection area to this direction is identified by using distance information to the target as well. In this case the closest intersection area to the main target is I1. Then the robot motion is triggered towards to I1 intersection area. The local target is assigned to upmost middle point in this area because of location of intersection area. Aim of selecting the upmost position of the intersection area (local target) is that providing the robot remains in the boundary of the intersection area. On the other hand, aim of the middle position of the point is that to provide a balanced distance between obstacles. The exact position of the local target may change according to the location of intersection area. Therefore, it may be leftmost, rightmost, lowermost and uppermost. However middle position is selected vertically or horizontally. It should be remembered that C1-C3 and C2-C4 intersection areas are horizontal and C1-C2 and C3-C4 intersection areas are vertical. Intersection areas have been shown with red rounded rectangles in Fig. 99.

The default robot position and path simulation under C4 camera is given in Fig. 100 (I) and (II), respectively. The selected f1, f2, … , f8 frames showing robot positions and angles from initial to the final position in Fig. 101. The WMR has reached to the defined position about 4.24s in 46 frames. So, it can be said that 10.80 frames per second are processed while storing and displaying data tasks are activated. A_L, A_R and A_T angle changes and velocity of wheels during the control process are graphically

demonstrated in Fig. 102. The formed paths by simulation (blue) and real robot (red) are shown in Fig. 103. The path length is found as 442.51px in total for simulation and 423.45px in total for real experiment.

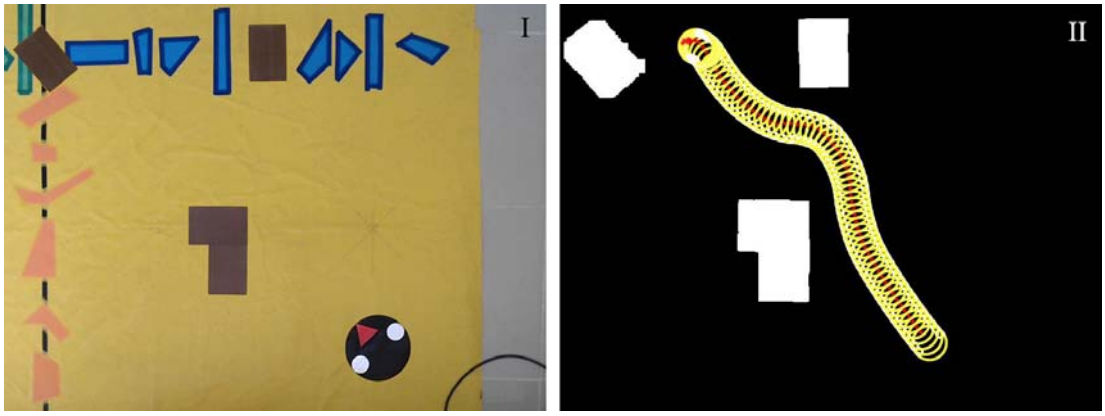

Fig. 100. (I) Camera 4 (C4) coverage area (II) Simulated path under C4

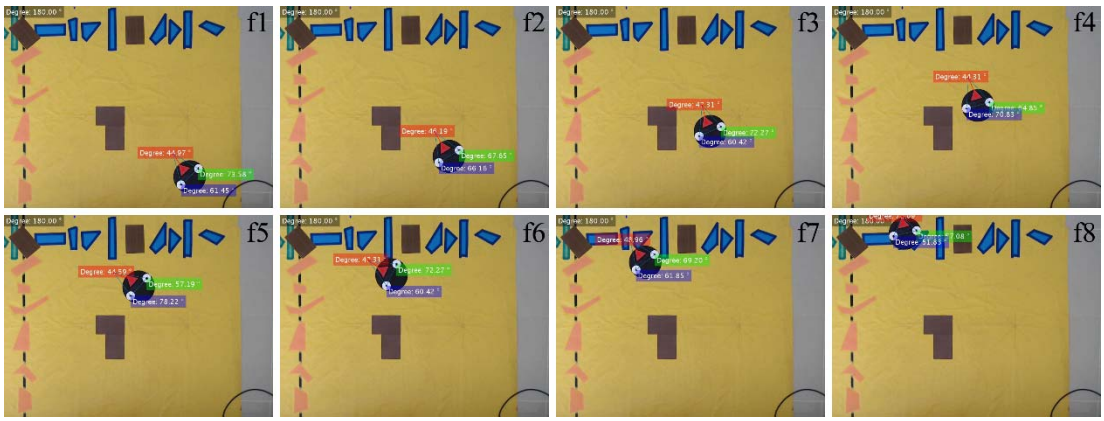

Fig. 101. Selected instance frames showing robot positions and angles

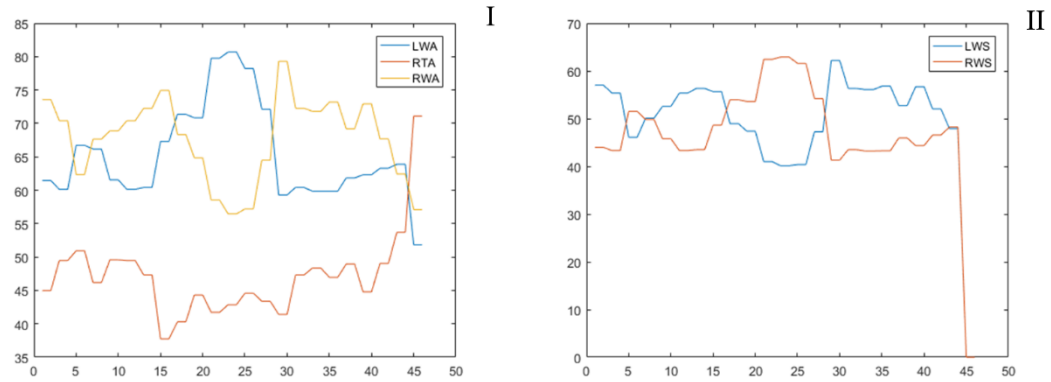

Fig. 102. (I) Angle changes of WMR control points (II) Velocity changes of WMR wheels

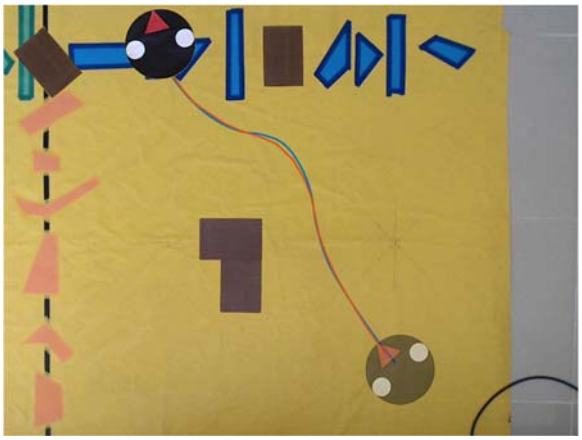

Fig. 103. Simulation path (blue) and Real path (red)

The default robot position and path simulation under C2 camera is given in Fig. 104. The selected $f1, f2, \ldots, f8$ frames showing robot positions and angles from initial to the final position in Fig. 105. The WMR has reached to the defined position about 2.12s in 26 frames. So, it can be said that 12,26 frames per second are processed while storing and displaying data tasks are activated. A_L, A_R and A_T angle changes and velocity of wheels during the control process are graphically demonstrated in Fig. 106. The formed paths by simulation (blue) and real robot (red) are shown in Fig. 107. The path length is found as 283.34px in total for simulation and 271.18px in total for real experiment.

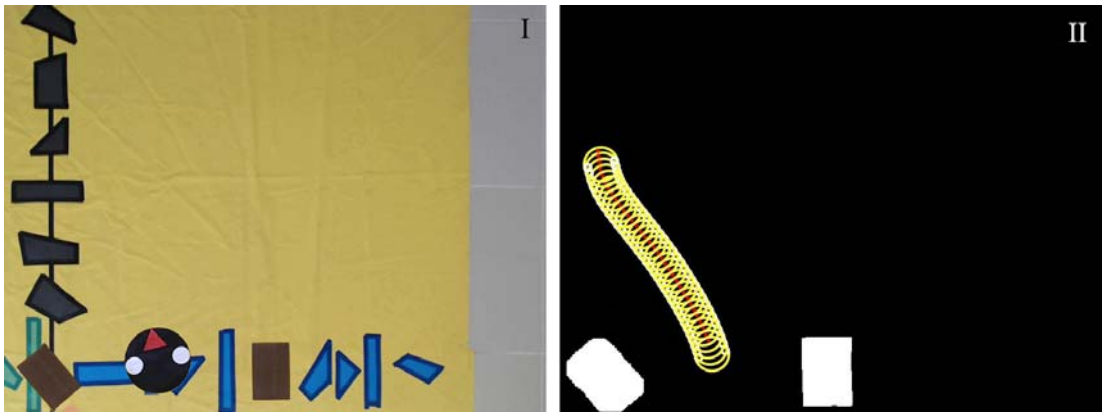

Fig. 104. (I) Camera 2 (C2) coverage area (II) Simulated path under C2

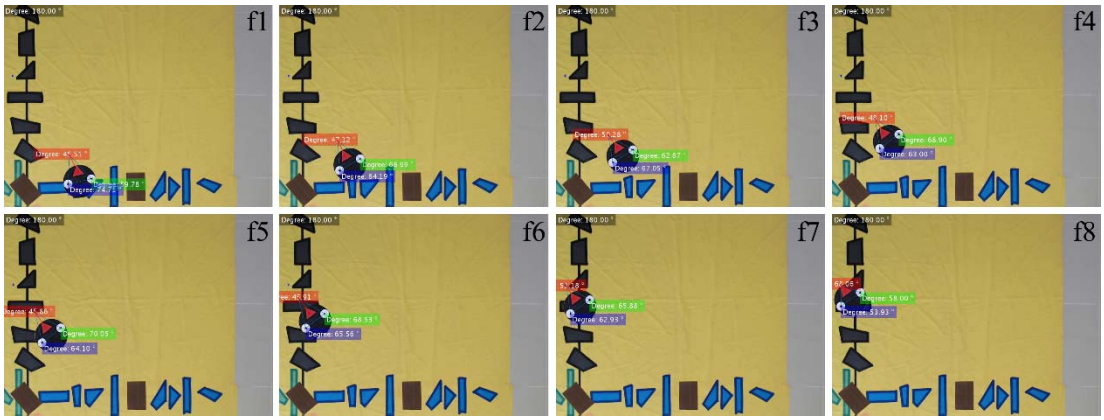

Fig. 105. Selected instance frames showing robot positions and angles

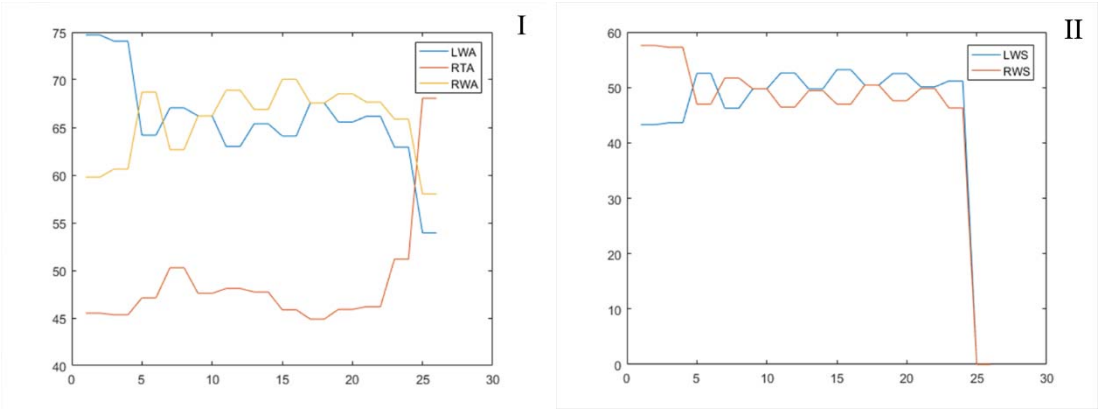

Fig. 106. (I) Angle changes of WMR control points (II) Velocity changes of WMR wheels

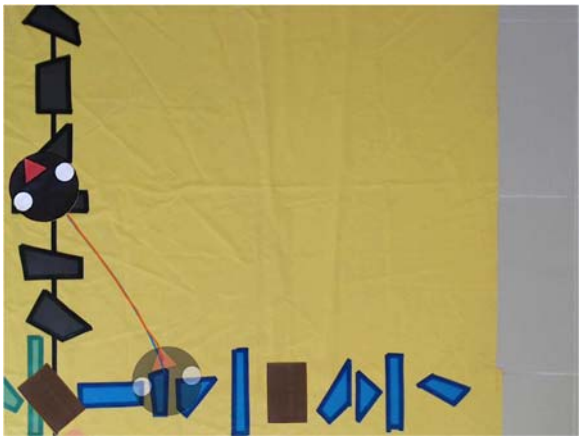

Fig. 107. Simulation path (blue) and Real path (red)

The default robot position and path simulation under C1 camera is given in Fig. 108. The selected $f1, f2, \ldots, f8$ frames showing robot positions and angles from initial to the final position in Fig. 109. The WMR has reached to the defined position about 3.62s in 42 frames. So, it can be said that 11,60 frames per second are processed while storing and displaying data tasks are activated. A_L, A_R and A_T angle changes and velocity of wheels during the control process are graphically demonstrated in Fig. 110. The formed paths by simulation (blue) and real robot (red) are shown in Fig. 111.

The path length is found as 354.19px in total for simulation and 332.83px in total for real experiment.

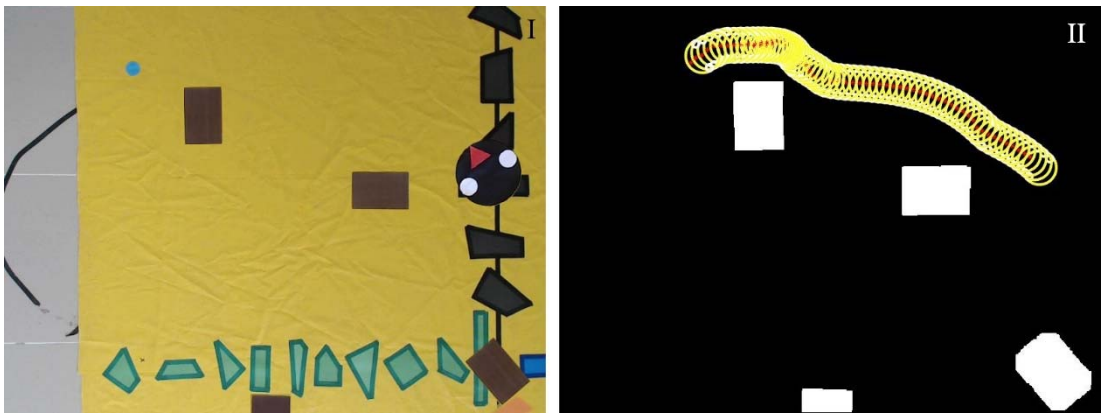

Fig. 108. (I) Camera 1 (C1) coverage area (II) Simulated path under C1

Fig. 109. Selected instance frames showing robot positions and angles

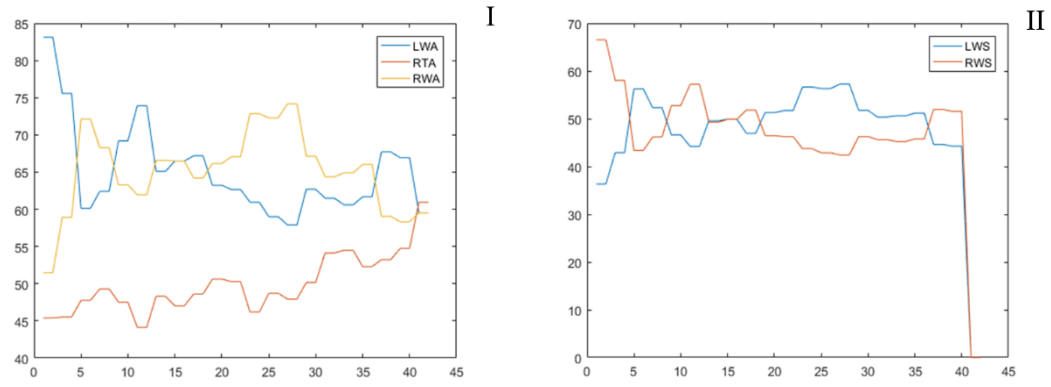

Fig. 110. (I) Angle changes of WMR control points (II) Velocity changes of WMR wheels

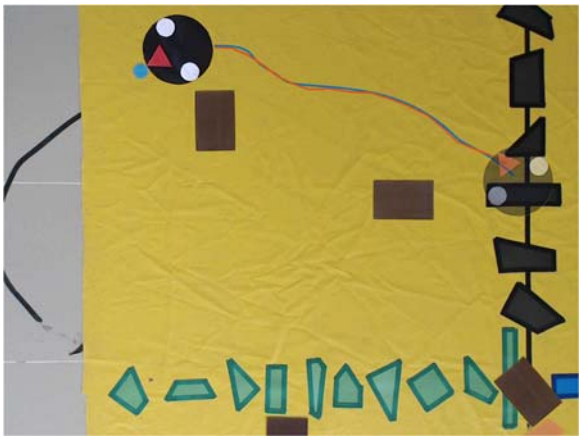

Fig. 111. Simulation path (blue) and Real path (red)

The experiment results have been summarized in Table 13. According to the obstacle alignment and configuration space specifications, utilized cameras may change. In other words, cameras with the different number and the different coverage areas can be utilized until the predefined target position is reached. Moreover, a camera may be utilized more than once to perform given control task(s). Comparing to the image stitching based multi-camera model, this pure multi-camera model can achieve better simulation and implementation times. Average simulation time is decreased from 11.7s to 10.46s with 10.6% gain and average implementation time is decreased from 12.07s to 9.71s with 19.55% gain. However, for all experiment configurations in pure model, except from Conf-3; path cost is increased about 3.01% from 1089.75 to 1122.56 for simulation and path cost is increased about 1.1% from 1049.95 to 1061.46 for implementation. Therefore, it can be said that simulation and implementation time of pure model is generally better than stitch-based model. On the other hand, simulation and implementation path cost of stitch-based model is mostly better than pure model. The main reason behind this situation is that the complete path model is extracted from the whole configuration space including all robot, target and obstacles in stitch-based model. However, path is partly extracted from local configuration space of related camera according to the robot position in pure model.

Table 13. Acquired time and cost values for different configurations

Experiment	Utilized Cameras	Simulation Time (s)	Implementation Time (s)	Simulation Path Cost (px)	Real Path Cost (px)
Conf-1	C4, C2, C1	9.98	9.07	1080.04	1027.46
Conf-2	C4, C3, C1	10.75	9.88	1185.63	1121.57
Conf-3	C4, C2, C1	10.64	10.17	1102.03	1035.36

5.4. The Main Influencers for Control Models

Illumination highly affects the threshold masks in the object detection process. Because of that both color features and color quantization methods are employed together to implement accurate object detection in the acquired images. If experiments are conducted under minor changing or constant light conditions, then masks operate without any error. The calibration of camera and image distortion affects data computations in processed image frames. The results are obtained with low accuracy without good and stable calibration and distortion. For example; a camera with fisheye lenses is not appropriate as a head imaging device without calibration. Calibration and distortion should be done according to the environment parameters.

The Gaussian and decision tree-based controllers have been exhibited an efficient and promising control task. It is easy to implement in visual based control designs. It can be used with both simple (angle, encoder etc.) and complex input parameters (depth, lidar information etc.). It has demonstrated a great consistency with potential field method. Several modifications may be required to apply it to other path planning methods.

Standard potential field method is sometimes insufficient to meet admissible safety and cost issues. Main reasons behind these are the local minimum, unstable oscillation, obstacle positions and so on. Safe path is created by dilation process on detected object in the most of studies. Geometric calculations also provide relative solutions to these problems. But eventually most effective and robust method is providing parameters which are changeable according to conditions of configuration space. Such mechanisms can be created with adaptive methodologies.

System hardware also influences performance of the visual based robot control. It should be emphasized that all the processed images are displayed and stored in real-time besides the stored WMR controller data. If the processing of displaying and storing the image data to the storage unit are deactivated, then images can be processed about 72.4% more performance averagely. There is no parallelization in control processes with accelerators like Graphic Processing Unit (GPU) processors. If GPU or CPU acceleration is exerted by parallelizing the existing control algorithms, then the performance can clearly be increased to better levels. Therefore, hardware is one of the most prominent factors in a real-time VBC system.

Multi-camera based visual control allows robot to be operated by the designed control system for large areas in interior spaces. The camera specifications should be same to acquire better efficiency and performance. Another factor is number of the cameras, but an external computer can cope with a certain number of cameras. Therefore, as the number of the camera increases the number of required computers will also increase. This situation is the main drawback of such systems. Illumination is a significant factor for multi-camera as it is in the single camera configuration. Each camera may be exposed different level of illumination. This difference may cause improper stitching process. Another option is managing the visual control process without images stitching. However, the path planning may emerge as a challenging issue.

5.5. A Multi Target Design with Load Balancing

5.5.1. System Design

Additional modules have been developed in order to determine the system performance in multiple targets and to suggest a load balancing system (LBS). There are multiple targets and two robots in the environment. The goal is to achieve a balanced distribution of workloads by considering the cost for both robots. As in the previous implementations, the configuration space image is obtained from the head camera.

The global positions of the robots and targets located on the input image of the configuration space are determined by the methods of color quantization and thresholding. The coordinates of the specified targets (or graph nodes) are kept in a matrix. The distance vectors from robots to nodes are calculated with these coordinates of targets and robots. The distance vectors are placed in a vector matrix table with the corresponding target ID and coordinate values. The number of vector matrix table will be calculated as much as the number of robots in the system. Two separate vector matrix tables are created for two robots. Then, the dimensions of the matrix space to be formed according to the number of targets in the environment are determined. The distance values of each target to both robots are compared according to the robot positions and the vector magnitudes in each vector matrix table. The target is assigned to the robot's navigation class according to the robot's proximity status to this target and the target having corresponding ID is deleted from both vector

matrix tables. This procedure has been repeated while the LBS has in both active and inactive status. Thus, the effect of load balancing on the navigation route has been investigated. The new navigation classes may also contain a different number of target elements according to the status of whether LBS is on or off. Fig. 112 shows the general operating steps of the system up to the creation of matrices (TMs) having target information for each robot. These matrices of targets include information about the targets to be navigated for each robot.

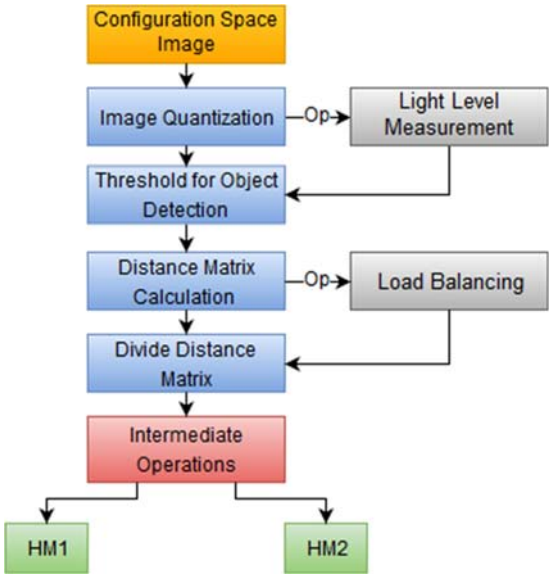

Fig. 112. Creating matrices holding target information

Fig. 113 illustrates the object identification steps for the components in the configuration space having multiple targets and two robots. All targets detected have been retained in a target matrix table.

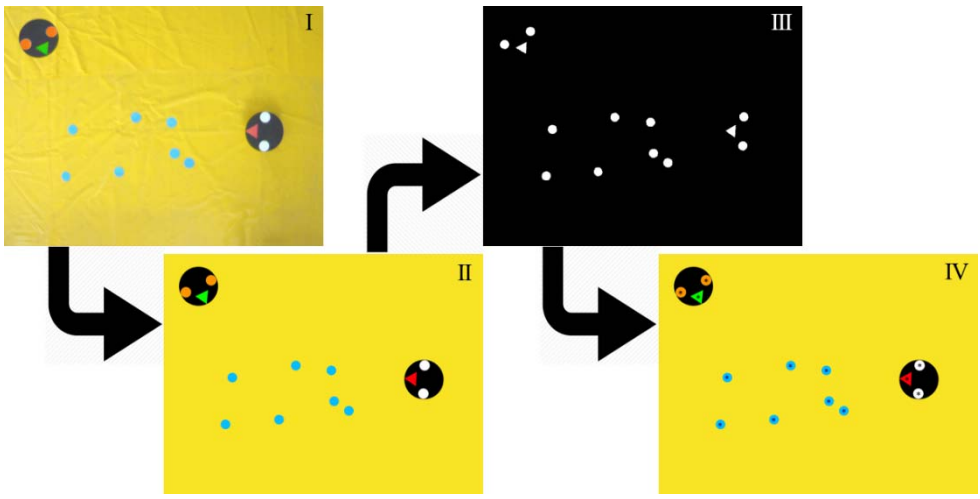

Fig. 113. Color-based component detection process: (I) Real-environment image, (II) Quantized image, (III) Binary map view of the environment, (IV) Detected components

5.5.2. Load Balancing System (LBS)

Load balancing is used to allocate the targets (workloads) in the environment in a balanced and effective manner among existing robots. By using the load balancing, the aim is to enable the robots to touring their given target areas faster and with less cost. The proposed load balancing method carries out the division of targets by considering two criteria. The first of these criteria is to distribute a balanced number of targets. For example; nine targets can be allocated as four and five targets between two robots in the configuration space. It is aimed to give the similar number of workloads to the robots. For this control, the distance of each target to the robots is checked and the assignment procedure is performed according to the proximity status, which is the first criterion of equilibrium. Table 14 below shows the first step of the load balancing algorithm applied according to the target parameters. This process takes place before the path plan is extracted.

Table 14. Load balancing algorithm based on number of targets

Input-1: Target location matrix – $\boldsymbol{KM}$
Input-2: Distance vector matrices – $\boldsymbol{MR1, MR2}$
Input-3: Number of targets – $\boldsymbol{HS}$
Equilibrium limit – $\boldsymbol{DS}$: $\boldsymbol{HS/2}$
If $\boldsymbol{HS \leq 0}$ then,
END
If $\boldsymbol{HS > 0}$ then,
Loop: $\boldsymbol{i}$=**0**; $\boldsymbol{i}$<$\boldsymbol{HS}$; $\boldsymbol{i}$++
If $\boldsymbol{MR1_i < MR2_i}$ then,
New **KM** Matrix**:** $\boldsymbol{KM_{R1} \leftarrow KM_i}$
If $\boldsymbol{MR1_i > MR2_i}$ then,
New **KM** Matrix**:** $\boldsymbol{KM_{R2} \leftarrow KM_i}$
Else if $\boldsymbol{MR1_i = MR2_i}$ then,
If $\boldsymbol{n(KM_{R1}) || n(KM_{R2}) \geq DS}$ then and
If $\boldsymbol{n(KM_{Rx}) < n(KM_{Ry})}$ $\boldsymbol{x, y \in \{1, 2\} \wedge x \neq y}$
$\boldsymbol{KM_{Rx}} \leftarrow$ Remaining targets
End Condition
End Loop
End Condition
Output: $\boldsymbol{KM_{R1}, KM_{R2}}$

According to the number of targets, LBS algorithm takes three inputs in the first stage. These parameters are the KM matrix which includes the position information of the targets, the MR1 and MR2 matrices which contain the distance data of each robot to all targets and the HS parameter which is the number of targets. If the number of targets (HS) is '1' or higher, then the iteration is triggered and run until HS value is reached. At each stage, the distance of each target to the robots is checked, and if the robot is close to the target, then the target is assigned to this robot.

In case of equality of distance values, the target is assigned to a random robot. Assignment of targets to the nearest robot will continue until the number of targets assigned to one of the robots is equal to the DS (half of the number of targets). When this value is reached, all remaining targets are assigned to the robot with fewer assigned targets. Thus, a target assignment process is performed between the robots by considering the number of targets. In the last case, one more or an equal number of targets are assigned to one of the robots. The KM_{R1} and KM_{R2} are newly created matrices of target positions. The $n(KM_{Rx})$ is the number of targets in the matrix. The x and y parameters are integers.

The second criterion for LBS is the traveling distances of the robots. The LBS algorithm checks this criterion after the acquisition of the cost of the path with the nearest neighbor or genetic algorithm. In this context, after the path plans are obtained in two classes having balanced number of targets, the path costs for each robot are calculated and compared. According to this cost, one or more of the workloads assigned to a robot can be reassigned to the other robot. The algorithm given in Table 15 performs this process after the path plan has been extracted.

Table 15. Load balancing algorithm according to path costs

Input-1: Shared target locations matrix – $\boldsymbol{KM_{R1}}$, $\boldsymbol{KM_{R2}}$
Input-2: Path plan costs for R1 and R2 – $\boldsymbol{YP_{R1}}$, $\boldsymbol{YP_{R2}}$
Input-3: Distance vector matrices – $\boldsymbol{MR1}$, $\boldsymbol{MR2}$
Absolute difference of path plan costs – $\boldsymbol{YP_{MF} = |YP_{R1} - YP_{R2}|}$
If $\boldsymbol{YP_{MF} < min(YP_{R1}, YP_{R2}) * 0,20}$ then,
END
Loop: while $\boldsymbol{YP_{MF} \geq min(YP_{R1}, YP_{R2}) * 0,20}$,
If $\boldsymbol{maks(YP_{R1}, YP_{R2}) = YP_{R1}}$ then,
$\boldsymbol{KM_{Rx} = KM_{R1} \wedge x = 1}$
$\boldsymbol{KM_{Ry} = KM_{R2} \wedge y = 2}$
$\boldsymbol{MR_x = MR1 \wedge MR_y = MR2}$
If $\boldsymbol{maks(YP_{R1}, YP_{R2}) = YP_{R2}}$ then,
$\boldsymbol{KM_{Rx} = KM_{R2} \wedge x = 2}$
$\boldsymbol{KM_{Ry} = KM_{R1} \wedge y = 1}$
$\boldsymbol{MR_x = MR2 \wedge MR_y = MR1}$
End Condition
$\boldsymbol{H_{Rx} = maks_d(KM_{Rx})}$,
New KM Matrices: $\boldsymbol{KM_{Ry} = KM_{Ry} \cup H_{Rx}}$, $\boldsymbol{KM_{Rx} = KM_{Rx} \backslash H_{Rx}}$
New Path Cost: $\boldsymbol{YP_{Ry} = d(KM_{Ry}) \in \{MR_y\}}$, **NN||GA**
New Path Cost: $\boldsymbol{YP_{Rx} = d(KM_{Rx}) \in \{MR_x\}}$, **NN||GA**
End Loop
End Condition
Output: $\boldsymbol{YP_{Ry}}$, $\boldsymbol{YP_{Rx}}$ and $\boldsymbol{KM_{Ry}}$, $\boldsymbol{KM_{Rx}}$

The KM_{R1}, KM_{R2} shared target positions matrices, YP_{R1}, YP_{R2} matrices including path costs for each robot and $MR1$, $MR2$ distance data for each robot to all targets are

given as input to LBS algorithm according to the path costs. If the absolute value of the difference between the cost of two path plans is less than 20% of the smallest path cost YP_{MF}, then the algorithm is terminated. In case of the difference is large than the 20%, the parameters for the high path costs are assigned to the parameters KM_{Rx}, MR_x and x, and parameters for the low path costs are assigned to the parameters KM_{Ry}, MR_y and y. After this assignment, the farthest node distance value to the robot position is determined for the path having the largest distance value and it is assigned to the H_{Rx} parameter. In the next step, the H_{Rx} target is added to the new target location matrix KM_{Ry} and the target H_{Rx} is subtracted from the KM_{Rx} matrix. New path costs are found by re-calculating the KM_{Rx} and KM_{Ry} distance matrices of the targets with GA and NN. This target addition/subtraction and reassignment process continues until the threshold value between the path costs is reached. If this threshold value is not reached, then the algorithm terminates and the nearest value close to the threshold value is considered as the solution. In order to see the efficiency of load balancing, the LBS module is integrated into the system as an optional plug-in.

5.5.3. Nearest Neighbor Method

The nearest neighbor method uses the distance between the target nodes to create a path in a simple way. The start position node can be determined by the algorithm or by external selection. After selecting a starting position, the distances between this node and other target nodes are calculated. This distance calculation is done by using the Euclidean distance equation. In the next step, all distances from the starting node to the others are compared and the node with the smallest distance from the starting point is selected as the second target to be visited. This new node is reassigned as the start node. The previous node is removed from the navigation matrix. All previous steps are repeated for this new node. This path extraction process continues until all variables in the dataset are processed in the same way. When all node visits are completed, a path is obtained that visits all nodes. The resulting pathway may be the least cost path, but the closest neighbor method generally produces an acceptable level of cost in the TSP (Travelling Salesman Problem). The closest neighbor method has been used as its default specifications in this study. Fig. 114 shows the working diagram for the NN method.

Fig. 114. Nearest Neighbor (NN) working diagram

5.5.4. Genetic Algorithm (GA)

Genetic algorithm is a method that is inspired by nature and makes problem solution closer to the best solution by performing a search and optimization process on the target dataset. It tries to reach a holistic solution based on the principle of survival of the best, while conducting the search process in multidimensional and complex space. The genetic algorithm generates a set of solutions unlike a single solution to the problem. Since it works with nature inspiration, selection attempts to further improve the data population with crossing and mutation steps. The algorithm is stopped when the maximum number of iterations is reached. The method tries to make the best choice for the current situation through the fitness function. It produces effective, efficient and useful solutions when traditional optimization methods do not give good or expected results. Fig. 115 shows the working scheme of the algorithm. It is commonly used to solve the TSP. The calculation of the fitness value is based on the distance values between these object nodes. As a result of tests for maximum iteration, an upper limit has been determined empirically.

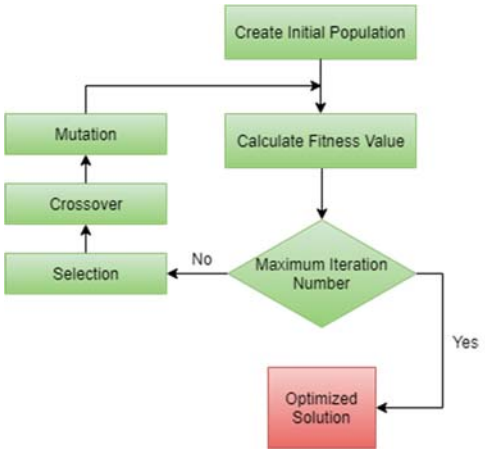

Fig. 115. Genetic Algorithm (GA) working diagram

5.5.5. Findings and Observations

In order to see the performance values of the proposed system, a configuration space with a total of two different target layouts including 8 and 24 targets have been used. Two robots (R1 - Red, R2 - Green) share the tasks in the system. Robot positions are arranged so that they are opposite directions. The distribution of targets has been set in three forms; random (R), stacked to one side (S) and collective (C). The 12 different experiments have been performed in total according to three target distributions, two different target numbers and activation status of LBS (active/inactive).

In the task sharing, the proximity of the targets to the robots is considered. In this case, it can be said that load balancing is ignored. When LBS is activated, it is ensured that the total distance values of the paths followed by the robots are brought closer together. In some studies, it is tried to provide load balancing by distributing the targets in balanced numbers to the robots. However, it is not provided an efficient solution by considering only the number of targets. Since, the costs of the paths may be very different from each other. For this reason, both number of targets and cost of traveling criteria are taken into account for load balancing in this study.

The experiment environment is as shown in Fig. 116 for the configuration of 8 targets. In the figure, there is a random distribution in part I, in section II is piled to the right side, and in section III there is a collective distribution.

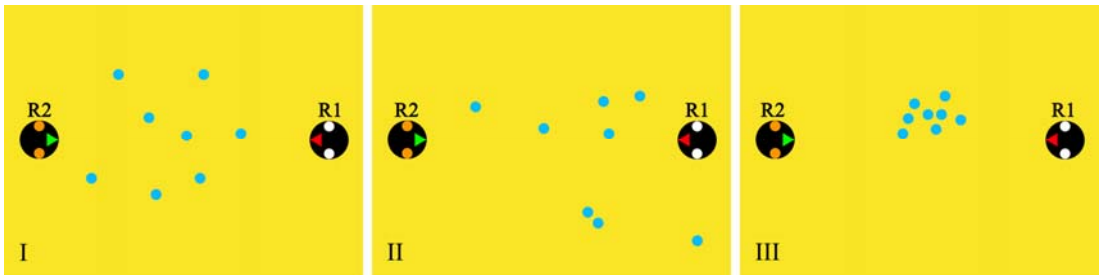

Fig. 116. Different distribution configurations of '8' targets in different positions

The path plans obtained for the 8 targets with nearest neighbor and genetic algorithm methods are given in Fig. 117. The path shown by red has been obtained by the NN method, while the path shown by blue has been obtained by GA. When the path plans are examined, it is observed that the genetic algorithm creates similar path plans with the nearest neighbor and differentiates in some sections. This difference leads to cost differentiation in path costs. This is due to the fact that the methodology

of the methods is different in the evaluation process. On the other hand, the number of iterations for GA in 8 target configurations are set to 30.

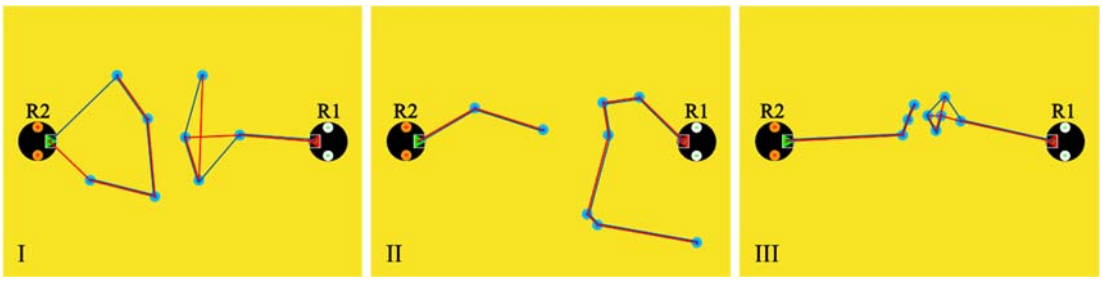

Fig. 117. Path plans for 8 targets with NN (red) and GA (blue) methods

Fig. 118 shows the paths plans created when LBS is open. As it can be seen, an equal number of targets are assigned to each robot in all distributions. After this assignment, navigation plans have been extracted between targets with NN and GA. In the next step, the second part of the algorithm is run if the difference between the distances of path plans exceeds the threshold value. Since this threshold is not exceeded in these distributions, path plans are considered to be efficient.

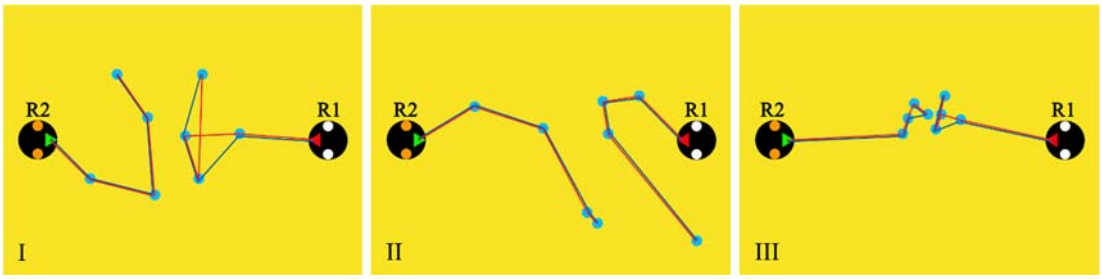

Fig. 118. Acquired path plans for '8' targets (LBS open)

The experiment environment is as shown in Fig. 119 for the configuration with 24 targets. In the figure, there is a random distribution in part I, in section II is piled to the right side, and in section III there is a collective distribution.

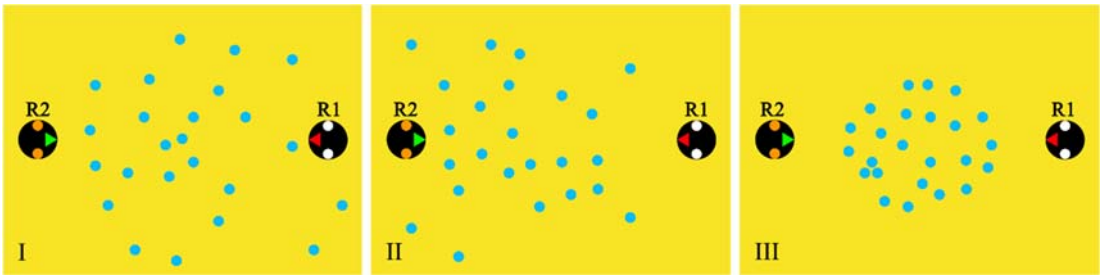

Fig. 119. Different distribution configurations of '24' targets in different positions

The path plans obtained for the 24 targets with the NN and GA methods are given in Fig. 120. The path shown by red has been obtained by the NN method, while the path shown by blue has been obtained by the GA method. Similarly, although there is

a similarity ratio of the paths obtained in the same way, there are also sections in which they differ. The number of GA iterations has been set to 60 for 24 targets. As the number of targets increases, the minimum number of iterations needed also increases. Increasing the number of iterations can provide better results in the GA method, but the algorithmic performance decreases. In experiments (I) and (III), where the distribution shows a homogeneous characteristic, it is seen that similar or close number of targets are assigned to R1 and R2 robots. On the other hand, in the experiment (II) in the middle side, 8 targets have been assigned to the R1 robot while 16 targets have been assigned to the R2 robot. This causes an unbalanced workload distribution between the robots.

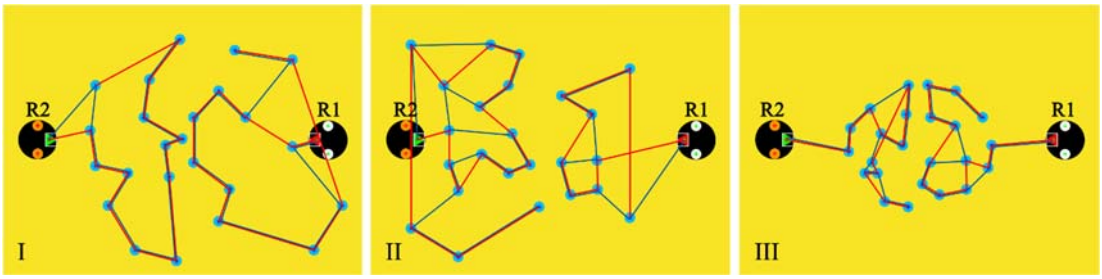

Fig. 120. Path plans for 24 targets with NN (red) and GA (blue) methods

In Fig. 121, only one target has been reassigned to the R1 robot from R2 robot by executing the LBS algorithm, (I). The distribution has been performed again according to the number of targets and path cost in the experiment, (II). A more balanced target assignment has been made to the robots. There is no change due to the fact that the targets are already balanced and the difference between path costs is below the threshold value in the experiment, (III).

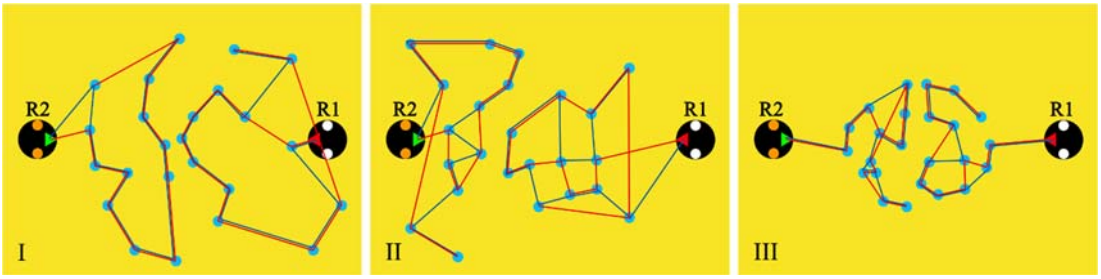

Fig. 121. Acquired path plans for ‘24’ targets (LBS open)

Distributing the target tasks to the robots with load balancing by considering the number of targets and the closeness of the targets to the robots ensures that the paths obtained do not cross. This structure minimizes the negative situations of robots such as waiting and disturbing each other.

The path costs are given in Table 16 for the R1 robot and in Table 17 for the R2 robot. The obtained path costs by NN and GA methods have been given in all target numbers and distribution configurations while LBS is inactive. Table data provides basic data to see the individual workloads of robots. The GA method has generally generated paths lesser cost than the NN method except for a few configurations. On the other hand, there are also plans having same cost with the NN and GA methods. As the number of targets increases, path costs generally increase in both methods. According to the distribution of the targets, it has been observed that the cost is higher in the random distribution and the cost is lower in the collective distribution.

Table 16. Path costs (px) obtained in experiments for R1 - LBS closed (LBS-C)

Exp. Name	Distributions					
	R		S		C	
	NN	GA	NN	GA	NN	GA
8 Target	624	593	734	734	401	395
24 Target	**1428**	1342	**1098**	**985**	**837**	**842**

Table 17. Path costs (px) obtained in experiments for R2 - LBS closed (LBS-C)

Exp. Name	Distributions					
	R		S		C	
	NN	GA	NN	GA	NN	GA
8 Target	565	588	**305**	**305**	362	349
24 Target	1349	1318	1820	1746	878	866

The path costs are given in Table 18 for the R1 robot and in

Table 19 for the R2 robot. The obtained path costs by NN and GA methods have been given in all target numbers and distribution configurations while LBS is active. Similarly, the GA method has given better results than the NN method when the load balancing is active. The difference between path distances has been further reduced by LBS. The results given in bold text mean that better results are obtained when the load balancing is active. It can be said that LBS generally provides better results for task sharing, except for a few cases.

Table 18. Path costs (px) obtained in experiments for R1 - LBS open (LBS-O)

Exp. Name	Distributions					
	R		S		C	
	NN	GA	NN	GA	NN	GA
8 Target	**624**	**593**	**613**	**613**	**361**	**352**
24 Target	1436	**1302**	1495	1276	840	856

Table 19. Path costs (px) obtained in experiments for R2 - LBS open (LBS-O)

Exp. Name	Distributions					
	R		S		C	
	NN	GA	NN	GA	NN	GA
8 Target	**565**	**565**	546	546	**352**	**344**
24 Target	**1268**	**1218**	**1369**	**1209**	**798**	**781**

Table 20 shows the total workloads (costs) for each configuration. When the total workloads are examined, it is understood that GA method gives better results than NN method in most cases. There are also cases where the NN method equals to the GA in the total path costs. The GA method has ensured improvements in path costs from 0% to 13.24% compared to the NN method. On the other hand, when the effect of load balancing on the total path cost is examined, it has been observed that the LBS provides improvements in all other cases except for the 8 target tests where the distribution is stacked (S). In the case which could not improve the overall cost of the path, the workload has been given with the similar costs to the robots in the background and significant improvements have been achieved. This could provide better energy management. These results indicate that the higher the number of targets, the better the load balancing results. On the other hand, a more efficient working infrastructure has been built in terms of time and energy by providing similar number of workloads to robots.

Table 20. Total workload of robots in each configuration (total cost)

Exp. Name	Distributions					
	R		S		C	
	NN	GA	NN	GA	NN	GA
8 Target LBS-C	1189	1181	**1039**	**1039**	732	726
8 Target LBS-O	**1189**	**1158**	1159	1159	**713**	**696**
24 Target LBS-C	2777	2660	2918	2731	1715	1708
24 Target LBS-O	**2704**	**2520**	**2864**	**2485**	**1638**	**1637**

5.5.6. Results and Recommendations

In this study, the task sharing for robot, balanced load distribution, path plan extraction, multiple TSP problem and the necessary methods have been discussed on random, one side stacked and collective distributions with different target numbers. The main focus of this study is distributing targets with a balanced manner. The targets have been assigned to the robots efficiently in terms of both number and cost with LBS. According to the obtained results, it has been observed that LBS improves the default path plan in many different scenarios. LBS has produced efficient results

in terms of multiple targets and task sharing across multiple robots. In terms of cost, GA has showed more successful performance. In terms of speed, the NN method performed much better than the GA method. The main reason of this situation is that the GA method tries to improve the solutions obtained in each step by depending on the number of iterations. Both methods have showed advantages and disadvantages according to their usage areas and needs. A total of 12 different experiments have been performed according to two different target numbers, three different distributions and two different LBS status.

6. CONCLUSION AND FUTURE WORKS

The WMR motions has been successfully characterized in each image frame and it has reached to the target position with high accuracy. The developed control methods ensure more robust, flexible and simple controlling process and eliminate systematic and unsystematic errors. The designed control infrastructure is primarily suitable for interior spaces. For instance; forklift trucks can be efficiently controlled in a warehouse with VBC systems. Since external or internal distance sensors are not necessary, system cost will decrease by using the proposed method. Modelling and adapting it to any indoor environment are easy.

The developed method is the first method using parameters of detected objects to form and operate a dynamic structured triangle or graph, directly on the robot. Besides, as far as we know, this is the first study employing Gaussian function as a default mobile robot controller by modifying several parameters in real time. This is the first study which uses decision tree-based controller as a novel method in VBC systems. Firstly, we modeled and designed only the go-to-goal control task which is one of the major task components in robotic applications alongside the navigation, obstacle avoidance etc. tasks. Then we have designed adaptive potential field method for path planning and combined it with previously designed go-to-goal controllers. Ultimately, the multi-camera infrastructure design is successfully harmonized with proposed methods.

We plan to combine this study with a newly designed or available visual-based control systems by comparing experiment parameters. Additionally, we will integrate and test commonly known path planning approaches with our method. Ultimately, it is aimed to apply this study to configuration spaces with obstacles and multi targets.

Visual based control presents design ideas in another level for robotic applications. Eye-in-device (or internal sensor) applications generally requires calculation of distance parameters from 2D images by considering depth information. Therefore, it requires much complex processes as disadvantage. Eye-out-device (or external sensor) requires distance information with a good calibrated camera as well. But this configuration provides simple approaching to control process. The main disadvantage is that the camera is generally placed to a fixed position. Therefore, there is a coverage area for the camera.

In this thesis study, a novel Gaussian/Decision tree controller, adaptive artificial potential field methods and multi-camera configuration are proposed and they have been combined for a visual based control. By using the proposed methods WMR has reached to the target position for all configurations. A robot is admitted as a point mass in many studies. However, the WMR is admitted with its whole dimensions in both simulations and real implementations for this study. Moreover, we have touched key issues extensively and have made a wide literature search.

In this thesis study, a multi-camera model is proposed for VBC. It provides an expandable and scalable platform. Unlike the stereovision imaging, depth information is not used and it does not provide any remarkable advantage for this configuration. The mobile robot has successfully reached to the target position in each configuration. It has been focused to multi-camera model and path planning in the scope of this work. Therefore, we have used a basic color thresholding-based object detection method. We plan to use and investigate learning based object detection methods in our next studies. Ultimately, we will extensively focus these issues in later studies.

A model will be designed by preparing the necessary infrastructure to test an alternative graph-based method in Fig. 122. It will used to determine the feasible paths between the obstacles. After determining the obstacles in the obstacle-hosted environment, the corner points of the obstacles are extracted with an algorithm like Harris. Then circles whose diameter equal to the distance between these points are created between the closest corner points or between the corner points and the obstacle edges. The center points of these circles are found. Circles that are smaller in diameter than the diameter of the robot are eliminated. Paths that do not intersect the obstacles between the circles are drawn so that they coincide to the center of the circle. Next, there is a diagram showing the routes on which the robot can proceed. These paths between the circle centers are our edges and the centers of the circle are our nodes. At the last stage, the cost of cross sections between these nodes is calculated in length. Inter-node costs can be kept in an adjacency matrix. Ultimately, this graph will be the input for path planning, it will be used to find the shortest path.

In Fig. 122, the robot can go to the green nodes. However, it cannot go to the red nodes, which is mainly because the diameter of the circle around the node is smaller than the diameter of the robot. The graph in the figure is illustrated as an example. It

is aimed to make the path planning process steps more durable and smoother with additional improvements, techniques and heuristic methods.

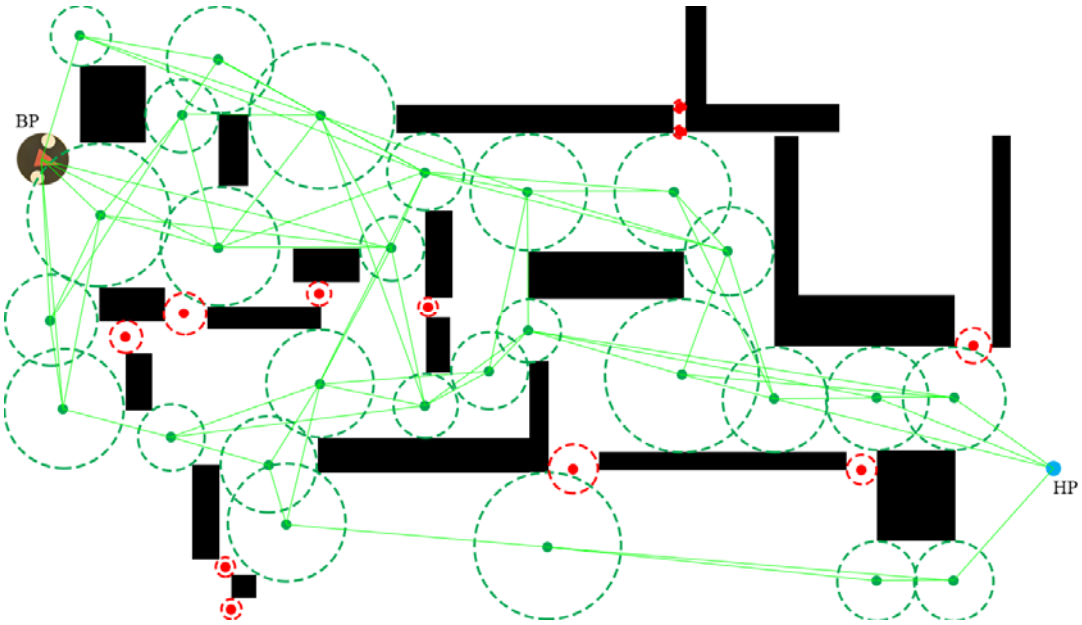

Fig. 122. Creating a graph-based path; The green nodes are the nodes to be gone, and the red nodes are the blind nodes. BP: Initial Position, HP: Target Position

Despite all the problems, visual based robot control under multi-camera surveillance is a young area which should be studied in-depth. According to the all knowledge and expertise acquired within this study, it can be clearly said that both eye-out-device and eye-in-device based visual control systems shows a promising future.

REFERENCES

[1] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin, *Iterative feedback tuning: Theory and applications*, **IEEE Control Syst. Mag.**, 18:4 (1998) 26–41.

[2] Z. Y. Zhao, M. Tomizuka, and S. Isaka, *Fuzzy Gain Scheduling of PID Controllers*, **IEEE Trans. Syst. Man Cybern.**, 23:5 (1993) 1392–1398.

[3] P. Shah and S. Agashe, *Review of fractional PID controller*, 38 (2016) 29–41.

[4] A. A. Voda and I. D. Landau, *A method for the auto-calibration of PID controllers*, 31:1 (1995) 41–53.

[5] P. S. Londhe, Y. Singh, M. Santhakumar, B. M. Patre, and L. M. Waghmare, *Robust nonlinear PID-like fuzzy logic control of a planar parallel (2PRP-PPR) manipulator*, **ISA Trans.**, 63 (2016) 218–232.

[6] H. Choset *et al.*, *Principles of Robot Motion*, 2005 .

[7] G. Dudek and M. Jenkin, *Computational principles of mobile robotics*, Cambridge University Press, 2010 .

[8] B. Siciliano and O. Khatib, Robotics and the Handbook, *Springer Handbook of Robotics*. Cham: Springer International Publishing, pp. 1–10, 2016.

[9] N. J. Cowan, J. D. Weingarten, and D. E. Koditschek, *Visual servoing via navigation functions*, **IEEE Trans. Robot. Autom.**, 18:4 (2002) 521–533.

[10] F. Chaumette and S. Hutchinson, *Visual servo control. I. Basic approaches*, **IEEE Robot. & Autom. Mag.**, 13:4 (2006) 82–90.

[11] F. Chaumette and S. Hutchinson, *Visual servo control. II. Advanced approaches [Tutorial]*, **IEEE Robot. Autom. Mag.**, 14:1 (2007) 109–118.

[12] A. J. Koivo and N. Houshangi, *Real-time vision feedback for servoing robotic manipulator with self-tuning controller*, **IEEE Trans. Syst. Man. Cybern.**, 21:1 (1991) 134–142.

[13] K. Hosoda and M. Asada, *Versatile visual servoing without knowledge of true Jacobian*, Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94), (1994) pp.186–193.

[14] J. Wenger, *Automotive radar - status and perspectives*, IEEE Compound Semiconductor Integrated Circuit Symposium, 2005. CSIC '05., (2005) pp.4 pp.

[15] T. Bailey and H. Durrant-Whyte, *Simultaneous localization and mapping (SLAM): Part I*, **IEEE Robot. Autom. Mag.**, 13:3 (2006) 108–117.

[16] A. Eidehall, J. Pohl, F. Gustafsson, and J. Ekmark, *Toward Autonomous Collision Avoidance by Steering*, **IEEE Trans. Intell. Transp. Syst.**, 8:1 (2007) 84–94.

[17] H. Khalajzadeh, C. Dadkhah, and M. Mansouri, *A review on applicability of expert system in designing and control of autonomous cars*, The Fourth International Workshop on Advanced Computational Intelligence, (2011) pp.280–285.

[18] N. Cao and A. F. Lynch, *Inner–Outer Loop Control for Quadrotor UAVs With*

Input and State Constraints, **IEEE Trans. Control Syst. Technol.**, 24:5 (2016) 1797–1804.

[19] T. Ryan and H. J. Kim, *Probabilistic Correspondence in Video Sequences for Efficient State Estimation and Autonomous Flight*, **IEEE Trans. Robot.**, 32:1 (2016) 99–112.

[20] V. Lippiello *et al.*, *Hybrid Visual Servoing With Hierarchical Task Composition for Aerial Manipulation*, **IEEE Robot. Autom. Lett.**, 1:1 (2016) 259–266.

[21] B. Siciliano and O. Khatib, *Handsbook of Robotics*, Springer-Verlag, 2008 .

[22] S. M. LaValle and S. M., *Planning algorithms*, Cambridge University Press, 2006 .

[23] J. N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, **IEEE Trans. Automat. Contr.**, 40:9 (1995) 1528–1538.

[24] T. H. Cormen, T. H. Cormen, R. L. Rivest, and C. E. Leiserson, *Introduction to algorithms*, MIT Press, 2001 .

[25] P. Hart, N. Nilsson, and B. Raphael, *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*, **IEEE Trans. Syst. Sci. Cybern.**, 4:2 (1968) 100–107.

[26] A. Stentz, *Optimal and efficient path planning for partially-known environments*, Proceedings of the 1994 IEEE International Conference on Robotics and Automation, (1994) pp.3310–3317.

[27] S. M. Lavalle and S. M. Lavalle, *Rapidly-Exploring Random Trees: A New Tool for Path Planning*, (1998).

[28] S. M. LaValle and J. J. Kuffner, *Randomized kinodynamic planning*, Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C), (1999) pp.473–479.

[29] C. Y. Lee, *An Algorithm for Path Connections and Its Applications*, **IEEE Trans. Electron. Comput.**, EC-10:3 (1961) 346–365.

[30] J. Anderson and S. Mohan, *Sequential Coding Algorithms: A Survey and Cost Analysis*, **IEEE Trans. Commun.**, 32:2 (1984) 169–176.

[31] E. Rimon and D. E. Koditschek, *Exact robot navigation using artificial potential functions*, **IEEE Trans. Robot. Autom.**, 8:5 (1992) 501–518.

[32] Z. Ziaei, R. Oftadeh, and J. Mattila, *Global path planning with obstacle avoidance for omnidirectional mobile robot using overhead camera*, 2014 IEEE International Conference on Mechatronics and Automation, (2014) pp.697–704.

[33] E. Johnson, E. Olson, and C. Boonthum-Denecke, *Robot localization using overhead camera and LEDs*, Florida Artificial Intelligence Research Society Conference, (2012) pp.524–526.

[34] C.-H. L. Chen and M.-F. R. Lee, *Global path planning in mobile robot using omnidirectional camera*, 2011 International Conference on Consumer Electronics, Communications and Networks (CECNet), (2011) pp.4986–4989.

[35] Y. Mezouar and F. Chaumette, *Path planning for robust image-based control*,

IEEE Trans. Robot. Autom., 18:4 (2002) 534–549.

[36] A. Breitenmoser, L. Kneip, and R. Siegwart, *A monocular vision-based system for 6D relative robot localization*, 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, (2011) pp.79–85.

[37] S. R. Bista, P. R. Giordano, and F. Chaumette, *Appearance-Based Indoor Navigation by IBVS Using Line Segments*, **IEEE Robot. Autom. Lett.**, 1:1 (2016) 423–430.

[38] Q. Bateux and E. Marchand, *Histograms-Based Visual Servoing*, **IEEE Robot. Autom. Lett.**, 2:1 (2017) 80–87.

[39] B. Espiau, F. Chaumette, and P. Rives, A New Approach to Visual Servoing in Robotics, *IEEE Transactions on Robotics and Automation*, 8, :3. pp. 313–326, 1992.

[40] J. Pauli, *Learning-Based Robot Vision*, Springer Berlin Heidelberg, 2001 .

[41] Y. Zhao, L. Gong, Y. Huang, and C. Liu, *A review of key techniques of vision-based control for harvesting robot*, **Comput. Electron. Agric.**, 127 (2016) 311–323.

[42] E. Donmez, A. F. Kocamaz, and M. Dirik, *Robot control with graph based edge measure in real time image frames*, 2016 24th Signal Processing and Communication Application Conference (SIU), (2016) pp.1789–1792.

[43] M. Dirik, A. F. Kocamaz, and E. Donmez, *Vision-based decision tree controller design method sensorless application by using angle knowledge*, 2016 24th Signal Processing and Communication Application Conference (SIU), (2016) pp.1849–1852.

[44] F. Martinelli, *A Robot Localization System Combining RSSI and Phase Shift in UHF-RFID Signals*, **IEEE Trans. Control Syst. Technol.**, 23:5 (2015) 1782–1796.

[45] E. A. Elsheikh, M. A. El-Bardini, and M. A. Fkirin, *Practical path planning and path following for a non-holonomic mobile robot based on visual servoing*, 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference, (2016) pp.401–406.

[46] Fujie Wang, Lulu Song, and Zhi Liu, *Image-based visual servoing control for robot manipulator with actuator backlash*, **2016 3rd Int. Conf. Inf. Cybern. Comput. Soc. Syst.**, :1 (2016) 272–276.

[47] X. Zhang, Y. Fang, B. Li, and J. Wang, *Visual Servoing of Nonholonomic Mobile Robots With Uncalibrated Camera-to-Robot Parameters*, **IEEE Trans. Ind. Electron.**, 64:1 (2017) 390–400.

[48] A. Elfes, *Sonar-based real-world mapping and navigation*, **IEEE J. Robot. Autom.**, 3:3 (1987) 249–265.

[49] A. Elfes, *Using occupancy grids for mobile robot perception and navigation*, **Computer (Long. Beach. Calif).**, 22:6 (1989) 46–57.

[50] J. Borenstein and Y. Koren, *Real-time obstacle avoidance for fast mobile robots*, **IEEE Trans. Syst. Man. Cybern.**, 19:5 (1989) 1179–1187.

[51] J. Borenstein and Y. Koren, *The vector field histogram-fast obstacle*

avoidance for mobile robots, **IEEE Trans. Robot. Autom.**, 7:3 (1991) 278–288.

[52] R. M. Murray and S. S. Sastry, *Nonholonomic motion planning: steering using sinusoids*, **IEEE Trans. Automat. Contr.**, 38:5 (1993) 700–716.

[53] J.-P. Laumond, P. E. Jacobs, M. Taix, and R. M. Murray, *A motion planner for nonholonomic mobile robots*, **IEEE Trans. Robot. Autom.**, 10:5 (1994) 577–593.

[54] R. Fierro and F. L. Lewis, *Control of a nonholonomic mobile robot using neural networks*, **IEEE Trans. Neural Networks**, 9:4 (1998) 589–600.

[55] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, *Monte Carlo localization for mobile robots*, Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C), (1999) pp.1322–1328.

[56] J. J. Kuffner and S. M. LaValle, *RRT-connect: An efficient approach to single-query path planning*, Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), (2000) pp.995–1001.

[57] F. Arambula Cosío and M. A. Padilla Castañeda, *Autonomous robot navigation using adaptive potential fields*, **Math. Comput. Model.**, 40:9–10 (2004) 1141–1156.

[58] T. Bailey and H. Durrant-Whyte, *Simultaneous localization and mapping (SLAM): part II*, **IEEE Robot. Autom. Mag.**, 13:3 (2006) 108–117.

[59] Z. Xu, R. Hess, and K. Schilling, *Constraints of Potential Field for Obstacle Avoidance on Car-like Mobile Robots*, **IFAC Proc. Vol.**, 45:4 (2012) 169–175.

[60] B. Kovács, G. Szayer, F. Tajti, M. Burdelis, and P. Korondi, *A novel potential field method for path planning of mobile robots by adapting animal motion attributes*, **Rob. Auton. Syst.**, 82 (2016) 24–34.

[61] M. Guerra, D. Efimov, G. Zheng, and W. Perruquetti, *Avoiding local minima in the potential field method using input-to-state stability*, **Control Eng. Pract.**, 55 (2016) 174–184.

[62] D. Jia, M. Wermelinger, R. Diethelm, P. Krusi, and M. Hutter, *Coverage path planning for legged robots in unknown environments*, 2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), (2016) pp.68–73.

[63] D. J. Bennet and C. R. McInnes, *Distributed control of multi-robot systems using bifurcating potential fields*, **Rob. Auton. Syst.**, 58:3 (2010) 256–264.

[64] R. A. F. Romero, E. Prestes, M. A. P. Idiart, and G. Faria, *Locally oriented potential field for controlling multi-robots*, **Commun. Nonlinear Sci. Numer. Simul.**, 17:12 (2012) 4664–4671.

[65] Y. Yan and Y. Li, *Mobile robot autonomous path planning based on fuzzy logic and filter smoothing in dynamic environment*, 2016 12th World Congress on Intelligent Control and Automation (WCICA), (2016) pp.1479–1484.

[66] P. K. Das, H. S. Behera, P. K. Jena, and B. K. Panigrahi, *Multi-robot path planning in a dynamic environment using improved gravitational search*

algorithm, **J. Electr. Syst. Inf. Technol.**, 3:2 (2016) 295–313.

[67] O. Montiel, U. Orozco-Rosas, and R. Sepúlveda, *Path planning for mobile robots using Bacterial Potential Field for avoiding static and dynamic obstacles*, **Expert Syst. Appl.**, 42:12 (2015) 5177–5191.

[68] D. H. Santos, A. P. F. Negreiros, J. E. A. Jacobo, L. M. G. Goncalves, A. G. Silva Junior, and J. M. V. B. S. Silva, *Short-Term Path Planning for High-Level Navigation Control of N-Boat - The Sailboat Robot*, 2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR), (2016) pp.211–216.

[69] J. Tan, L. Zhao, Y. Wang, Y. Zhang, and L. Li, *The 3D Path Planning Based on A* Algorithm and Artificial Potential Field for the Rotary-Wing Flying Robot*, 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), (2016) pp.551–556.

[70] E. Donmez, A. F. Kocamaz, and M. Dirik, *Visual based path planning with adaptive artificial potential field*, 2017 25th Signal Processing and Communications Applications Conference (SIU), (2017) pp.1–4.

[71] M. Dirik, A. F. Kocamaz, and E. Donmez, *Static path planning based on visual servoing via fuzzy logic*, 2017 25th Signal Processing and Communications Applications Conference (SIU), (2017) pp.1–4.

[72] E. Donmez, A. F. Kocamaz, and M. Dirik, *Bi-RRT path extraction and curve fitting smooth with visual based configuration space mapping*, 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), (2017) pp.1–5.

[73] M. Dirik, A. F. Kocamaz, and E. Donmez, *Visual servoing based path planning for wheeled mobile robot in obstacle environments*, 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), (2017) pp.1–5.

[74] S. Kamarry, L. Molina, E. A. N. Carvalho, and E. O. Freire, *Compact RRT: A New Approach for Guided Sampling Applied to Environment Representation and Path Planning in Mobile Robotics*, 2015 12th Latin American Robotics Symposium and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR), (2015) pp.259–264.

[75] Kunwook Lee, Ja Choon Koo, Hyouk Ryeol Choi, and Hyungpil Moon, *An RRT* path planning for kinematically constrained hyper-redundant inpipe robot*, 2015 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), (2015) pp.121–128.

[76] E. Shan, B. Dai, J. Song, and Z. Sun, *A Dynamic RRT Path Planning Algorithm Based on B-Spline*, 2009 Second International Symposium on Computational Intelligence and Design, (2009) pp.25–29.

[77] N. A. Melchior and R. Simmons, *Particle RRT for Path Planning with Uncertainty*, Proceedings 2007 IEEE International Conference on Robotics and Automation, (2007) pp.1617–1624.

[78] R. Heβ, T. Lindeholz, D. Eck, and K. Schilling, *RRTCAP∗ - RRT∗ Controller and Planner - Simultaneous Motion and Planning*, 48:10 (2015) 52–57.

[79] P. Muñoz, M. D. R-Moreno, and B. Castaño, *3Dana: A path planning*

algorithm for surface robotics, **Eng. Appl. Artif. Intell.**, 60 (2017) 175–192.

[80] E. Malis, F. Chaumette, and S. Boudet, *Multi-cameras visual servoing*, Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), pp.3183–3188.

[81] V. Lippiello, B. Siciliano, and L. Villani, *Eye-in-Hand/Eye-to-Hand Multi-Camera Visual Servoing*, Proceedings of the 44th IEEE Conference on Decision and Control, pp.5354–5359.

[82] L. Qiu, Q. Song, J. Lei, Y. Yu, and Y. Ge, *Multi-Camera Based Robot Visual Servoing System*, 2006 International Conference on Mechatronics and Automation, (2006) pp.1509–1514.

[83] Yuta Yoshihata, Kei Watanabe, and Yasushi Iwatani, *Multi-camera visual servoing of a micro helicopter under occlusions*, 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, (2007) pp.2615–2620.

[84] Y. Iwatani, Kohou, and K. Hashimoto, *Multi-camera visual servoing of multiple micro helicopters*, 2008 SICE Annual Conference, (2008) pp.2432–2435.

[85] B. Weber and K. Kuhnlenz, *Visual servoing using triangulation with an omnidirectional multi-camera system*, 2010 11th International Conference on Control Automation Robotics & Vision, (2010) pp.1440–1445.

[86] O. Kermorgant and F. Chaumette, *Multi-sensor data fusion in sensor-based control: Application to multi-camera visual servoing*, 2011 IEEE International Conference on Robotics and Automation, (2011) pp.4518–4523.

[87] E. A. Elsheikh, M. A. El-Bardini, and M. A. Fkirin, *Dynamic path planning and decentralized FLC path following implementation for WMR based on visual servoing*, 2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC), (2016) pp.1–7.

[88] H. Aliakbarpour, O. Tahri, and H. Araujo, *Visual servoing of mobile robots using non-central catadioptric cameras*, **Rob. Auton. Syst.**, 62:11 (2014) 1613–1622.

[89] K. Ahlin, B. Joffe, A. P. Hu, G. McMurray, and N. Sadegh, *Autonomous Leaf Picking Using Deep Learning and Visual-Servoing*, 49:16 (2016) 177–183.

[90] J. P. Alepuz, M. R. Emami, and J. Pomares, *Direct image-based visual servoing of free-floating space manipulators*, **Aerosp. Sci. Technol.**, 55 (2016) 1–9.

[91] I. Kolmanovsky and N. H. McClamroch, *Developments in nonholonomic control problems*, **IEEE Control Syst. Mag.**, 15:6 (1995) 20–36.

[92] T. Weerakoon, K. Ishii, and A. A. F. Nassiraei, *An Artificial Potential Field Based Mobile Robot Navigation Method To Prevent From Deadlock*, **J. Artif. Intell. Soft Comput. Res.**, 5:3 (2015) 189–203.

[93] D. Scaramuzza, A. Martinelli, and R. Siegwart, *A toolbox for easily calibrating omnidirectional cameras*, **IEEE Int. Conf. Intell. Robot. Syst.**, (2006) 5695–5701.

[94] R. C. Gonzalez and R. E. Woods, *Digital image processing*, Pearson Prentice

Hall, 2008 .

[95] N. Paragios, Y. Chen, and O. Faugeras, *Handbook of mathematical models in computer vision*, Springer, 2006 .

[96] C. Mota, J. Gomes, and M. I. A. Cavalcante, *Optimal image quantization, perception and the median cut algorithm*, **An. Acad. Bras. Cienc.**, 73:3 (2001).

[97] G. Dudek and M. Jenkin, *Computational principles of mobile robotics*, Cambridge University Press, 2010 .

[98] F. Chaumette and S. Hutchinson, *Visual servo control. I. Basic approaches*, **IEEE Robot. Autom. Mag.**, 13:4 (2006) 82–90.

[99] F. Chaumette and S. Hutchinson, *Visual servo control. II. Advanced approaches [Tutorial]*, **IEEE Robot. Autom. Mag.**, 14:1 (2007) 109–118.

[100] M. Brown and D. G. Lowe, *Recognising panoramas*, Proceedings Ninth IEEE International Conference on Computer Vision, (2003) pp.1218–1225 vol.2.

[101] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, *Speeded-Up Robust Features (SURF)*, **Comput. Vis. Image Underst.**, 110:3 (2008) 346–359.

[102] G. Lowe, *SIFT - The Scale Invariant Feature Transform*, **Int. J.**, 2 (2004) 91–110.

CIRCULLAUM VITAE

Name, Family-Name: *Emrah Dönmez*

Birthplace and Date: *Malatya – 1987*

Address: *İnönü University Faculty of Engineering Computer Engineering Department Robotic Laboratory[1], Malatya Technopark Administrative Department[2]*

E-Mail: *emrah.donmez@inonu.edu.tr[1], emrahdonmez@msn.com[2]*

Undergraduate Degree: *Computer Science – Suleyman Demirel University* - 2009

Master Degree: *Electronic-Computer Science – SDU* - 2011

PhD Course Period: *Computer Engineering Department – ITU*

Professional Experience: *Substitute Lecturer (2009-2010), Research Assistant (2010-2016), Academic Expert (2017-2018), Academician (2018-Now) Project Manager (2017-Now)*

Publication List:

A. Articles published in internationally acclaimed journals

Dönmez E., A. Kocamaz F., "Multi-Camera Configured Vision Based Mobile Robot Control with Path Planning", Arabian Journal for Science and Engineering, (2018). (SCI-E) – *Under Review*

Dönmez E., A. Kocamaz F., "Design of Mobile Robot Control Infrastructure Based on Decision Trees and Adaptive Potential Area Methods", Iranian Journal of Science and Technology, Transactions of Electrical Engineering, (2018). (SCI-E) – *Under Review*

Dönmez E., A. Kocamaz F., "Çoklu Hedeflerin Çoklu Robotlara Paylaştırılması İçin Bir Yük Dengeleme Sistemi", BEU Fen Bilimleri Dergisi, (2018). (TR-Dizin, ULAKBİM) – *Under Review*

Dönmez E., A. Kocamaz F., and Dirik M., "A Vision-Based Real-Time Mobile Robot Controller Design Based on Gaussian Function for Indoor Environment", Arab. J. Sci. Eng., (2017) 1–16. (SCI-E)

D. Emrah, Design of a Resource Management for GPGPU Supported Grid Computing**,** *Journal of Computer and Electrical Sciences (JCES),* Vol./Is. 1(1) (2016) pp. 39–48. ISSN: 2548-1304

Dönmez E., Özcan A., **"**Time Based Discovering Of Web User Patterns (Extended)". *International Journal of Advance Computational Engineering and Networking (IJACEN)*, 3(8), pp. 14-20, 2015

Aydoğan T., Gül K., **Dönmez E**., "Ultrasonik Sensör İle İki Boyutlu Haritalandırma Sistemi". *SDU International Journal of Technologic Sciences*, 1(1), pp. 1-9., 2009 (TÜBİTAK destekli lisans tezinden yapıldı)

B. Articles published in nationally-respected journals

-

C. Papers presented at international scientific conferences

Dönmez E. and Kocamaz A. F., "A Hog & Graph Based Human Segmentation from Video Sequences," *2018 International Artificial Intelligence and Data Processing Symp. (IDAP)*, Malatya, 2018, pp. 1-5.

Dönmez E. and Kocamaz A. F., "Multi Target Task Distribution and Path Planning for Multi-Agents," *2018 International Artificial Intelligence and Data Processing Symp. (IDAP)*, Malatya, 2018, pp. 1-8.

Dönmez E., Kocamaz A. F. and Dirik M., "Bi-RRT path extraction and curve fitting smooth with visual based configuration space mapping," *2017 International Artificial Intelligence and Data Processing Symp. (IDAP)*, Malatya, 2017, pp. 1-5.

Dirik M., Kocamaz A. F. and **Dönmez E.**, "Visual servoing based path planning for wheeled mobile robot in obstacle environments," *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, Malatya, 2017, pp. 1-5.

Toslak F., Kocamaz A.F. and **Dönmez E.**, "Designing and Developing a Voice Controlled Laser Printer to Code Microscope Slides Which is Used in Pathology Laboratories", *International Conference on Research in Education and Science (ICRES)*, Kuşadası/Aydın, Turkey, 2017, pp. 32-36.

Dönmez E., Kocamaz A. F. and Dirik M., "Visual based path planning with adaptive artificial potential field," *2017 25th Signal Processing and Communications Applications Conference (SIU)*, Antalya, Turkey, 2017, pp. 1-4.

Dirik M., Kocamaz A. F. and **Dönmez E.**, "Static path planning based on visual servoing via fuzzy logic," *2017 25th Signal Processing and Communications Applications Conference (SIU)*, Antalya, Turkey, 2017, pp. 1-4.

Dönmez E., Kocamaz A. F., Karcı A. "Melez (Bulut Ve Gönüllü) Küresel Hesaplama İçin Veri Güvenliği Ve Hesaplama Sisteminin İncelenmesi". *International Artificial Intelligence and Data Processing Symposium'16 (IDAP)*, pp. 561-568., 2016

Dirik M., Kocamaz A. F., **Dönmez E.**, "Vision-Based Decision Tree Controller Design Method Sensorless Application By Using Angle Knowledge". *24th Signal Processing and Communication Application Conference (SIU)*, pp. 1849-1852., 2016

Dönmez E., Kocamaz A. F., Dirik M. "Robot Control With Graph Based Edge Measure In Real Time Image Frames". *24th Signal Processing and Communication Application Conference (SIU)*, pp. 1789-1792., 2016

Dönmez E., Kocamaz A. F., Dirik M. "Robotic Positioning Method Design Through Image Based Virtual Path With Multi-Head Camera Infrastructure". *International Conference on Natural Science and Engineering (ICNASE'16)*, pp. 2278-2285., 2016

Kocamaz A. F., Dirik M., **Dönmez E.** "Head Camera-Based Nearest Neighborhood Relations Algorithm Optimization And The Application Of Collecting The Ping-Pong Ball". *International Conference on Natural Science and Engineering (ICNASE'16)*, 2385-2390., 2016

Dönmez E., Özcan A. "Time Based Discovering Of Web User Patterns To Optimize Web Sites And Hyperlinks". *ISERD - International Conference on Advances in Business Management and Information Technology (ICABMIT), 2015*

Dönmez E. "Design Of Security And Privacy Issues For Cloud Computing". *3rd International Symposium on Innovative Technologies in Engineering and Science*, 3(3), pp. 1840-1848., 2015

Dönmez E., Zadeh P. V. "A Modified Graph Based Approach For Leaf Segmentation With GPGPU Support". *23nd Signal Processing and Communications Applications Conference (SIU)*, pp. 1797-1800., 2015

Dönmez E., Kutlu A. "A Data Security System Design For Hybrid (Cloud & Volunteer) Global Computing". *6th International Conference on Information Security & Cryptology*, 1(6), pp. 3-9., 2013

D. Papers and Posters presented at national scientific conferences

Dönmez E., Gül K., "Ultrasonik Sensör ve Servo Motor ile İki Boyutlu (2D) Haritalandırma Sistemi", *SDÜ Ulusal Öğrenci Sempozyumu, 2010*

E. Projects

Dönmez E., vd., "İnönü Üniversitesi Girişimcilik Merkezi ve Ön-Kuluçka Kurulumu Projesi", Kalkınma Bakanlığı Cazibe Merkezleri Destekleme Programı (**CMDP**) Proje Koordinatörü, Proje Kodu: TRB1/2017/CMDP/001

Dirik M., **Dönmez E.**, Kocamaz A.F., "Engelli Ortamlarda Heterojen Dağılmış Hedefler için İşbirlikçi Çoklu Robotların Vizyon Tabanlı Görev Paylaşımlı Kontrolü", **TÜBİTAK 1002** Hızlı Destek Projesi (Quick Support Project) Project No: 116E568

Kocamaz A.F., **Dönmez E.**, Bayer H., "İplik Üretiminde Hata Kaynaklarının RFID Tekniği ile Belirlenmesi", Katılımlı Araştırma Projesi (KAP) – 1, Project ID: 1331

Kocamaz A.F., **Dönmez E.**, Okumuş F., "Yüksek Yük Kapasiteli Ürün Taşıyıcı Otonom Robot", Katılımlı Araştırma Projesi (KAP) – 2

Darwish A., **Dönmez E.**, Ören C. İ., "Hiteroskopi Balonu", Katılımlı Araştırma Projesi (KAP) – 3, Project ID: 1402

E. Reviews (Refereeing)

Journal of Intelligent & Robotic Systems (**SCI-E**), 16 Article

Intelligent Service Robotics (**SCI-E**), 5 Article

Journal of Parallel and Distributed Computing (**SCI**), 3 Article

Bitlis Eren University Journal of Science and Technology (**National**), 1 Article

Anatolian Science Journal (**International**), 1 Article