Team Description Paper for participation of Rimal Team in RoboCup Logistics League 2013

Mohammed Al Dabbous [1], Khaleel Al-Haboub, Abdulrahman Amri, Bilal Siddiqui, Hamzah Al Khadem

And

Sami El Ferik[2]

Rimal Team, Systems Engineering Department, King Fahd University of Petroleum & Minerals, 31261 Dhahran Saudi Arabia

([1]contact: midabbous@gmail.com, [2]Associate Professor: selferik@kfupm.edu.sa)

Abstract: This paper provides an introduction to the Rimal RoboCup Logistics team and Robotino robots. Robotino is one of the intelligent platform robots. The robot is flexible and easy to modify for different applications. Team members have experience in different aspects of robotics, controls, sensors technology, signal processing, programming, logic, and scheduling in order to provide an effective game strategy. The team is developing competitive group of robots that satisfies the RoboCup Logistics League. This paper describes the approach in developing, realizations, and future plan until the RoboCup 2013.

Keywords: Robotinos, Robocup Logistics, SLAM, Shortest Path, Image Processing.

1- Introduction

Rimal Team is the first team from Saudi Arabia established to participate in RoboCup Logistics League. Our team‘s open project is supported by King Fahd University of Petroleum and Minerals (KFUPM). The team has been created in September 2011 within the Systems Engineering Department, Control and Instrumentation group. Our aim is to share the knowledge with other research groups around the world and demonstrate our professionalism and commitment to excellence. With this in mind, we are ready to participate in Robocup Logistics using a group of Robotino Mobile Robots.

We have equipped and enhanced our Robotino Mobile Robots with many additional smart devices. We created intelligent methods in several areas including Image Processing, Navigation, Localization, Mapping, Controls and Object Recognition. This paper describes our system's hardware and the software components. In addition, we divided the project into 7 main tasks as follows:

- Programming Language & Dependencies
- Network Communications
- Motion Controller
- Sensors
- Image processing
- Game Logic
- Mapping

2- Programming Language & Dependencies

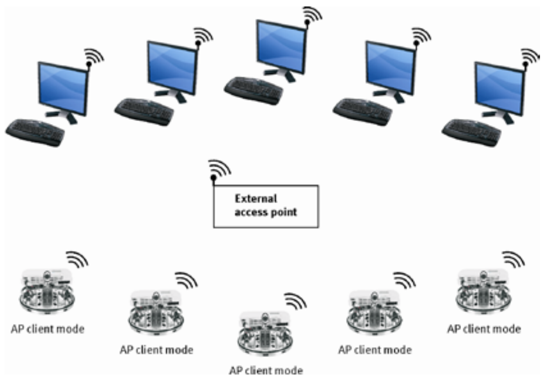
We are using C++ as main language to program the robots. Also, we are using some libraries to develop API in order to make the robot moving and having efficient interaction with the world. The libraries and the API that we are using are:

• **Robotino API 1**	This API contains all the functions needed to run the robot
• **QT library**	we use this library to create graphical user interface for testing.
• **OpenCV library**	OpenCV (Open Source Computer Vision Library)
• **Boost library**	this library contains some powerful algorithms.

In addition, Microsoft Visual Studio 2010 is our main compiler and Windows is our environment.

3- NETWORK COMMUNICATIONS

In order to make the robots communicate with each other, we set all the robots on client mode. Also, we used an Access Point Device to make all robots connect to it. Computers can also connect to that Access point. All robots and computers are connected to the same networks and work in Peer to Peer connection.



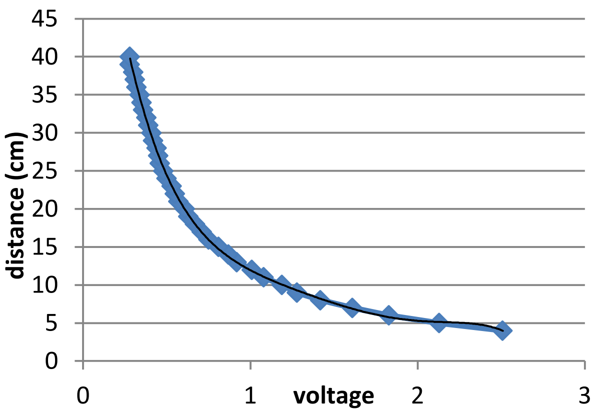
4- MOTION CONTROLLER

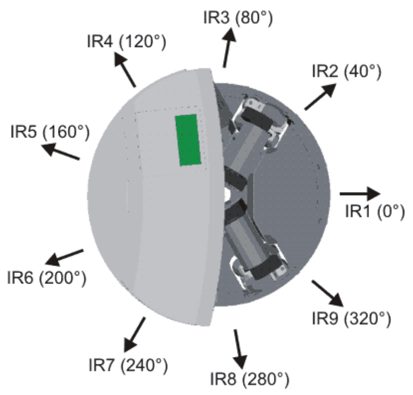
We use both liner controller and integral controller, to move the robots with margin error 3-6 cm. we are still testing and working on the controller to make it more accurate, and lower the margin error to 1-3 cm.

5- SENSORS

In the Robotino we have different type of sensors. We were able to implement and use the following:

- **Bumper**
  Bumper sensor is placed around the robot's circumference, to prevent it from crashing.
- **Distance Sensor**
  We have 9 IR sensors around the robot with 40˚ between each two sensors. We were able to convert the voltage that we get from the sensor into measurable distance. The accuracy that we got is 4-30 (cm). we are working now on collecting these reading from different sensors so we can detect the position of an object that is placed next to the robot.



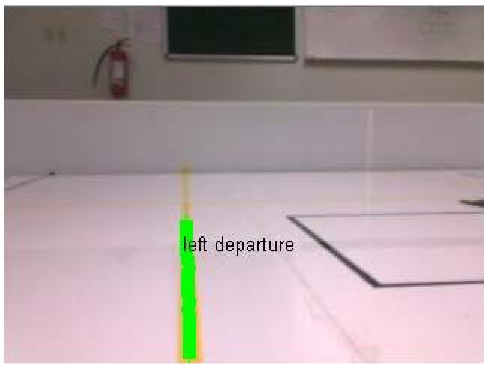


- **Optical Sensors**
  We have two optical sensors. These sensors can detect the black color on the floor.

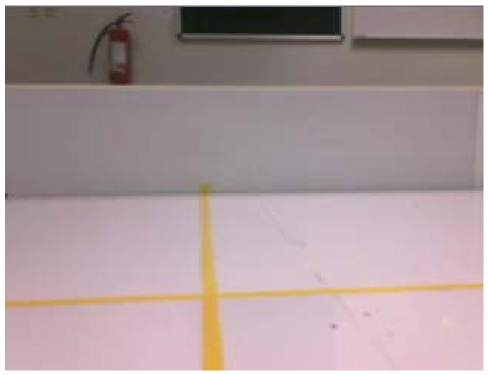
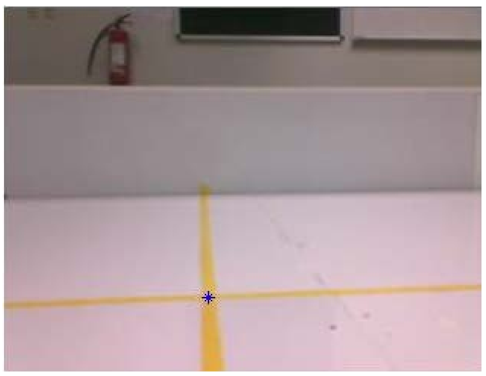
6- IMAGE PROCESSING

By using the ordinary camera attached to the Robotino and OpenCv library. We were able to perform the following:

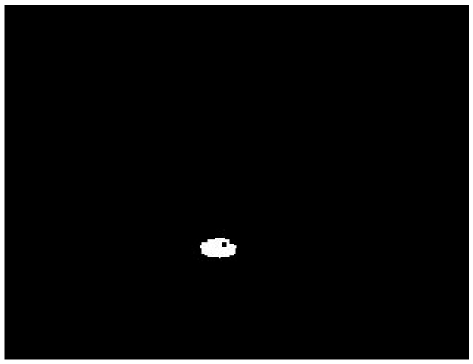
- **Line following**



- **Finding cross point**

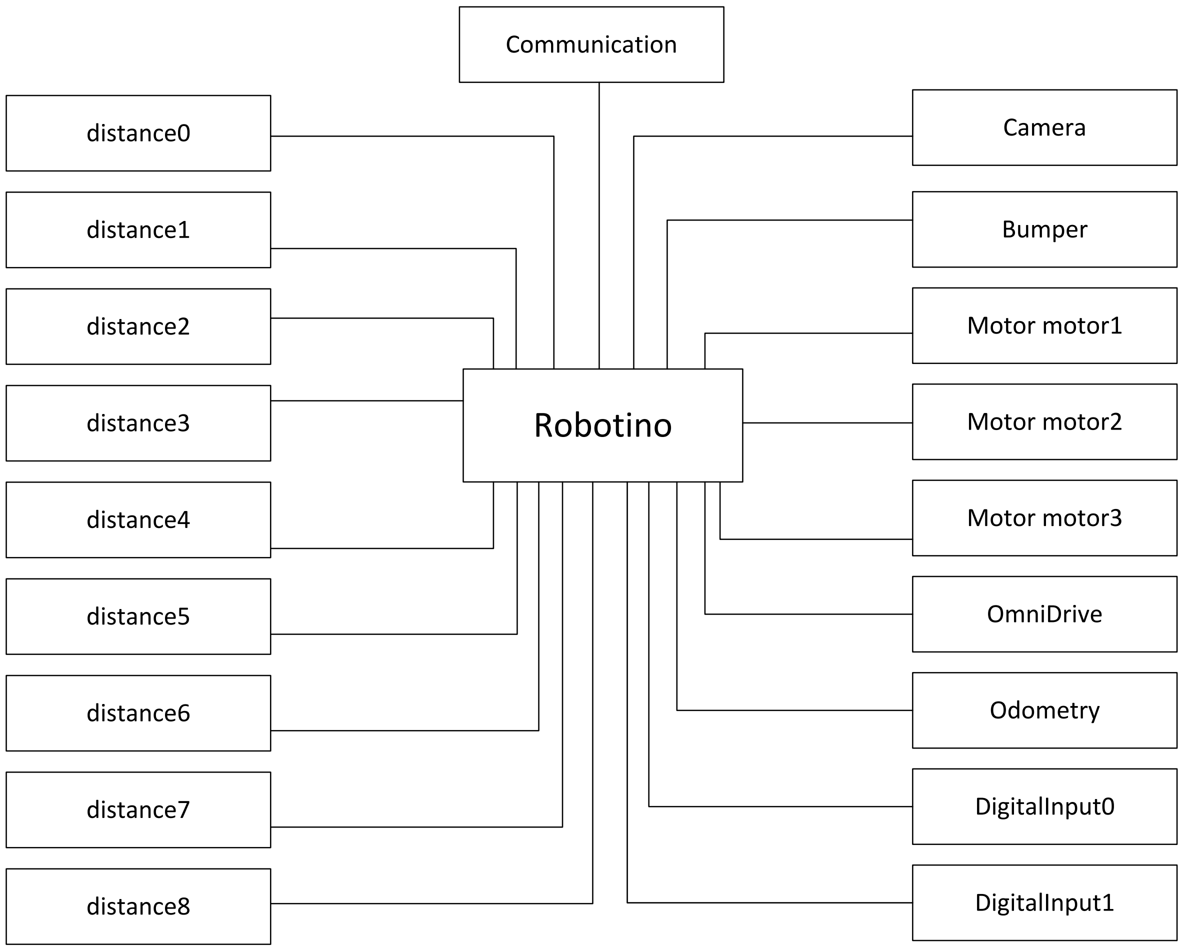
- **Detecting Colors**



7- ASSEMBLING THE PARTS

From software engineering perspective, we tried to make the software more modular, efficient, flexible and reusable. We grouped the codes in classes and methods so it can achieve our goals.

CLASS DIAGRAM

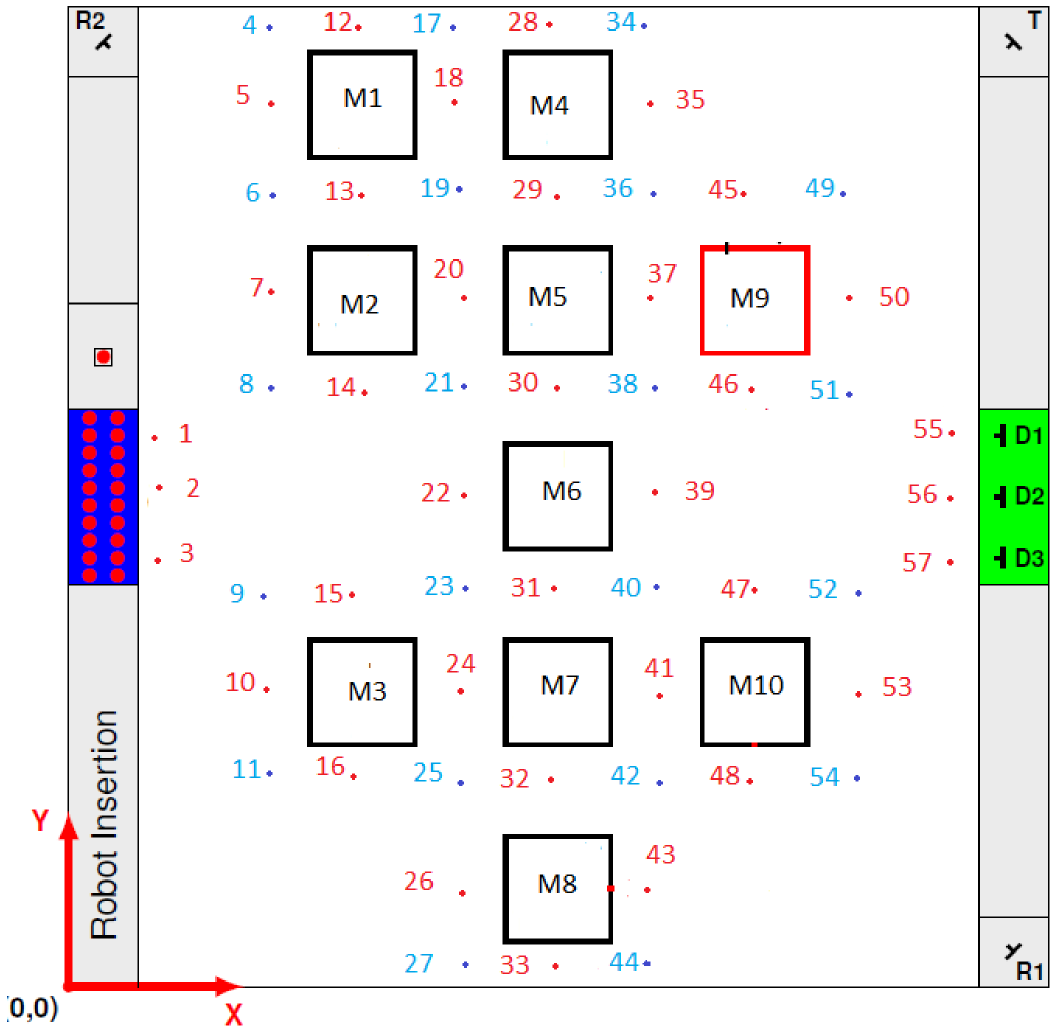


8- GAME LOGIC

This part is focusing in providing the shortest and best path algorithm for the robots, design best process plan, and create logic to avoid robot collision.

Shortest path

The problem is to find the shortest path between two red points.



Process Plan

The objective is to find best process plan that satisfy a given demand. For products P1,P2 and P3, the robot has 5 min to satisfy the demand. Every product has a certain processing sequence in order to deliver it.

Logic to avoid robot collision

In order to avoid robot collision, we examine the path of each robot and look if it has some intersections with other robots' paths. In addition, we evaluate the time that they may need to take the path.

Time estimated to finish the Logic

The time to do all the described work in the logic side is estimated to be1 week but to transform logic into code it might take up to 1 month.

9- MAPPING

Enable the robot to navigate itself in the field.

APPROACH DESCRIPTION

The problem was to make the robot move from point A to point B. One of the solutions proposed was to make the robot visualize the arena as a graph.
In the code level, the graph idea is planned to be implemented in a flexible way such that the robot would work on any graph whether it was in or out of the arena. An existing widely-known library called “Boost” is being used to help implement the code.

ABSTRACT SOLUTION APPROACH

1. Define the field as a graph that the robot understands
2. Place nodes on the graph
3. Unit testing(1)
4. Implement boost library(2)
5. Design an algorithm to link the nodes
6. Implement and integrate the algorithm
7. Provide interface for the solution to interact with the existing control program(3)
8. Testing

STEPS TO BE COMPLETED

1. Implement boost library
2. Implement and integrate the algorithm
3. Provide interface for the solution to interact with the existing control program
4. Testing

10- REFERENCES

- **Robotino API 1,** http://wiki.openrobotino.org/index.php?title=Downloads
- **QT library,** http://qt-project.org/downloads
- **OpenCV library,** http://www.opencv.org
- **Boost library,** http://www.boost.org

11- CONTACT US

If you have any further questions regarding our paper, then please do not hesitate to get in touch with us. For all questions related to our project and paper, your contact person is:

Mr. Mohammed I. Al Dabbous, e-mail: midabbous@gmail.com

[Feb 28, 2013]