ISSN 2347 - 3983
Volume 3, No.11 November 2015
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter013112015.pdf

Design and Implementation of Mobile Humanoid Robotic Arms

Mohammed Z. Al-Faiz[1], Abdulsalam A. Abdullah[2]

College of Information Engineering, Al-Nahrain University, Baghdad, Iraq mzalfaiz@ieee.org

[2]Department of Networks Engineering, Al-Nahrain University, Baghdad, Iraq,
salamahmed@coie.nahrainuniv.equ.iq

ABSTRACT

This paper proposes a system that designed to be used for picking or placing objects and to access areas that might be inaccessible to humans. The system comprises of three parts: the navigation algorithm, the robotic arms controlling interface and the live video streaming service. The shortest path between start and goal locations is computed via A* algorithm and forwarded to the mobile robot via a WLAN. The mobile robot uses Global Positioning System (GPS) and a digital compass for navigation. A Humanoid Robotic Arms (HRA) prototype with 5 Degrees-Of-Freedom (DOF) is designed. To control the HRA, the system uses a 3D human skeletal tracking method (implemented using Kinect sensor) and Inverse Kinematics. A Client/Server application is developed to transfer control signals and gather feedback signals from the robotic arms. The mobile robot reaches its final destination by a miss distance of less than three meters due to GPS inaccuracy. Depending on the feedback of the camera, a manual control from the PC base-station takes place to move it to the final destination. It successfully detects and avoids obstacles in its path using the 5 IR sensors. The system was successfully verified to pick and hold objects with one arm, but picking and holding objects with both arms was not verified due to insufficiency of the tracking capabilities (which implemented by the Kinect sensor and beyond the goal of this work).

Key words : A* algorithm, GPS, Humanoid Robotic Arms, Kinect.

1. INTRODUCTION

A robot is an automatic or semi-automatic machine capable of purposeful motion in response to its surroundings in an unstructured environment [1]. Mobile robots are an automatic machine which are able to move around in their environment. Unlike industrial robots, which usually consist of a jointed arm and end-effector (or gripper assembly) attached to a fixed surface [2]. Nowadays, robots are increasingly being integrated into working tasks to replace humans especially to perform repetitive tasks. The mobile robots are currently used in many fields of applications including office, military tasks, hospital operations, dangerous environment and agriculture. Besides, it might be difficult to the worker who must pick and place something that can affect badly. For example, dangerous chemical materials that cannot be picked by human and for the military such as defusing bombs that need robot to pick and place the bomb somewhere and for users who need robots to do pick and place items while sitting. Therefore a locomotion robot can replace human to do work. The robot is wireless controlled to ensure it can be used a long way from the user [3].

In many outdoor applications the robots can determine their coordinates by using the Global Positioning System (GPS). GPS is used to enable the robot as well the user to control the robot path as it provides the position, velocity, and timing information [4]. This paper presents a system that uses 3D Human Skeletal Tracking implemented by Kinect one device [5] shown in Figure 1 and inverse kinematics to control the humanoid robotic arms which integrated with a mobile robot that navigate based on GPS.

Figure 1: Microsoft Kinect One Sensor

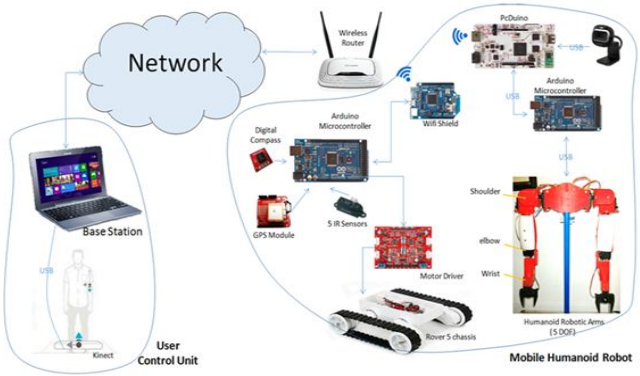


Figure 2: System Architecture

2. HUMANOID ROBOTIC ARMS DESIGN

The humanoid robotic arm was designed to have 5DOF. Each degree of freedom represents an angle of movement that can be implemented by human arm. A servo motor is used to carry out the movement of each angle ($Q_1 \rightarrow Q_5$). Pan and tilt mechanisms was used in order to allow servos to perform the required motion at each joint. The arm is consist of two parts, the upper part (shoulder to elbow) and the lower part (elbow to wrist). A gripper was added at the end of the arm to act as a hand with two fingers. The complete design of the humanoid robotic arm shown in Figure 3. The complete design was printed by using 3D plastic printer.

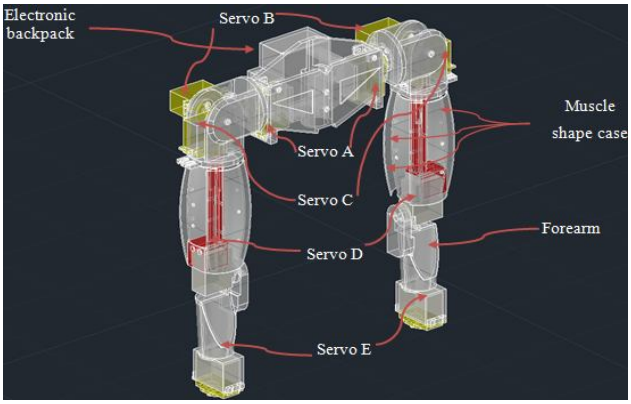


Figure 3: Complete Humanoid Robotic Design

3. SOFTWARE SYSTEM DESIGN

Software system is divided into three parts: Mobile Robot Navigation software and Robotic Arms software, and Video Streaming software as follows:

3.1 Mobile Robot Navigation Software

PC base station software is the main controller of the mobile robot unit. Interface was designed using MATLAB program version (R2015a), where it is responsible for three main task; the first task is getting the map of the search area. The second task is calculating the path for the mobile robot using A* algorithm and converting it to a series of latitude/longitude points. The third is transmitting this path to the mobile robot. Furthermore, base station can be used to control the mobile robot manually in the final stage (i.e. in the surrounding area of the target). Simplified scheme of algorithms principle is shown in Figure 4.

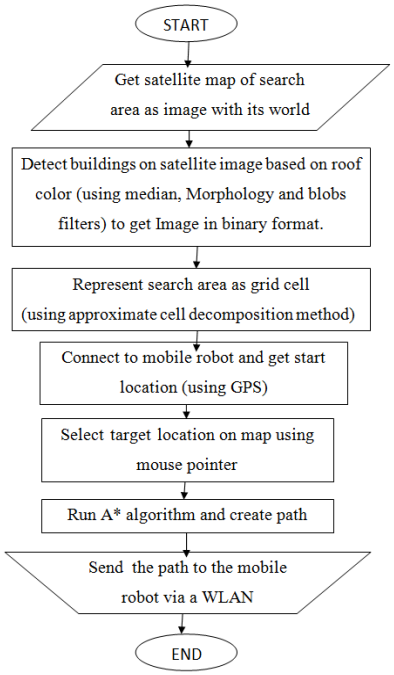


Figure 4: PC Base-Station Procedure for Mobile Robot Control

Mobile robot has two tasks to do, follow the path that is received from computer, detect and avoid obstacles in its path. It receives the path from computer as a number of points, each one will be considered as target point numbered as T1, T2…Ti….,Tgoal, for each iteration i, the robot finds the distance and the angle bearing between the current position that is received from GPS and the target point (Ti) according to 'haversine' formula (1) and bearing formula (2) [6].

$$a = \sin^2(\Delta lat/2) + \cos lat_1 \cdot \cos lat_2 \cdot \sin^2(\Delta lng/2)$$
$$\emptyset = 2 \cdot \operatorname{atan2}(\sqrt{a}, \sqrt{(1-a)})$$
$$d = Re \cdot \emptyset \quad (1)$$

$$\theta = \tan^{-1}(\sin \Delta lng \cdot \cos lat_2, \cos lat_1 \cdot \sin lat_2 - \sin lat_1 \cdot \cos lat_2 \cdot \cos \Delta lng) \quad (2)$$

where: $\emptyset$ is the angular distance in radians, *a* is the square of half the chord length between the points, d is distance, *lat* is latitude, *lng* is longitude, *Re* is earth's radius(mean radius = 6,371km), θ is bearing angle.

Two PID controllers are used, one decides which direction movement the mobile robot must take (turn right or left) to make its orientation straight to the target point and the other to make the mobile robot moves straight forward to its target point with minimum path perturbation depending on the difference between heading angle from the compass and the bearing angle, as shown in Figure 5.

After the mobile robot reaches the target point Ti then the next target point of path Ti+1 will be taken and the robot returns the same operation and so on until the end of path.

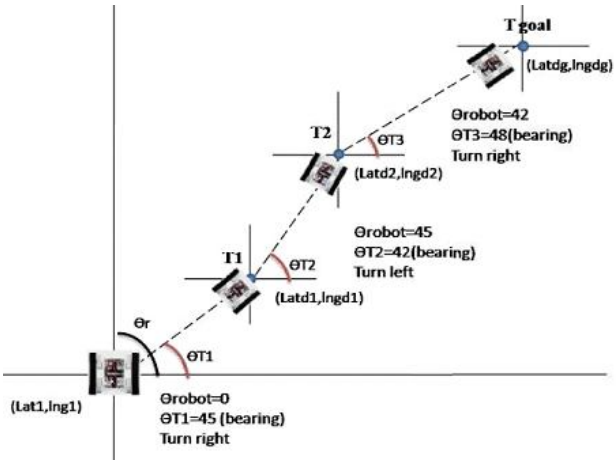


Figure 5: Mobile Robot Direction at Each Points of Path

To make Mobile robot detects and avoids obstacles, five IR sensors (sharp GP2Y0A21YK) are used for this purpose. These sensors are needed for front and side detection. These sensors are placed on the vehicles front side, left side, right side, front left side and front right side with a 45-degree deviation angle. Each sensor has an angle of view equal to 45 degree range finder and up to 80 cm. The mobile robot programmed to detect obstacles as far away 10 cm or less. When the mobile robot tries to move forward, if any obstacle prevents its movement it tries to turn its direction depends on the front left and right sensor, if any obstacle in right, it tries to

turn left and if any obstacle in left it tries to turn right, this turning on is a continuous process before frequently moving forward to the goal. The flowchart of the algorithm is shown at Figure 6 [7].

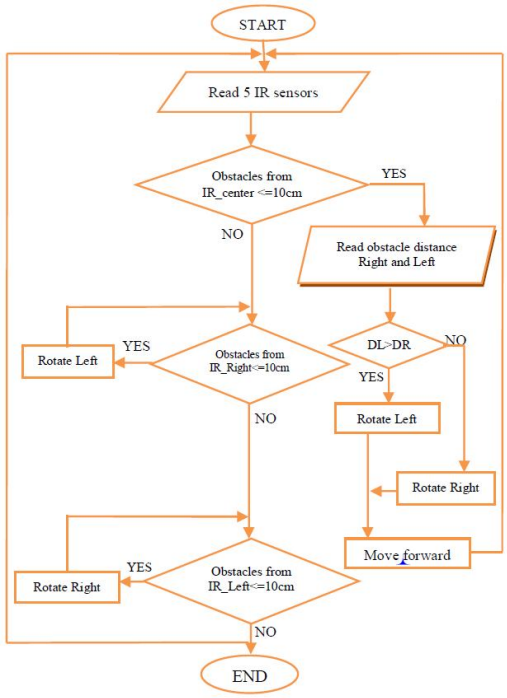


Figure 6: Flowchart of Obstacles Avoidance Process

3.2 Robotic Arms Controlling Software

Tracking and positioning of the human arm is carried out by continuously processing depth images of the user who is performing the arms motion to complete a robot manipulation task. Skeletal tracking algorithms are run on-board the Kinect sensor itself allowing the tracked skeleton data to be streamed to the computer through a simple stream declaration. The skeleton data being received from the stream contains the tracked skeleton joints of the user being viewed. These joints are returned as objects and their X, Y and Z positions in the field of view are extracted. Inverse kinematics computes the joint angles of a kinematics chain based on the position and orientation of the last kinematics segment. In this paper, the posture of the user arm is required to be observed, specifically of the user's right arm. As a human arm is consists of joints and bones, the joints was considered as 'start' and 'end' points and the bones was considered as vectors.

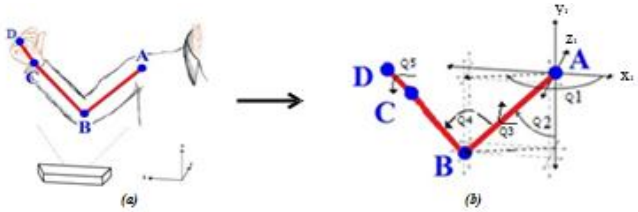


Figure 7: Inverse Kinematics Calculation

Figure 7(a). illustrates a scene where the Kinect is tracking a user's right arm. Points A, B, C and D represent the tracked shoulder, elbow, wrist joints and hand tip respectively. The joints A and B have been treated as 'start' and 'end' points respectively which have now formed a vector $\rho(\overrightarrow{AB})$, and in the same way we got $\rho(\overrightarrow{BC})$, $\rho(\overrightarrow{AC})$, and $\rho(\overrightarrow{CD})$ as shown in Figure 7(b). By using simple trigonometric methods, the angles $Q_1, Q_2, Q_3, Q_4 and\ Q_5$ can be calculated as shown in the following steps [8]:

Step 1: Finding vector $\underset{AB}{\rightarrow}, \underset{BC}{\rightarrow}, \underset{AC}{\rightarrow}$ and $\underset{CD}{\rightarrow}$.

$$\overrightarrow{AB} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (3)$$

$$\overrightarrow{BC} = \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2 + (z_3 - z_2)^2} \quad (4)$$

$$\overrightarrow{AC} = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2 + (z_3 - z_1)^2} \quad (5)$$

$$\overrightarrow{CD} = \sqrt{(x_4 - x_3)^2 + (y_4 - y_3)^2 + (z_4 - z_3)^2} \quad (6)$$

Step 2: Finding $Q_1, Q_2, Q_3, Q_4 and\ Q_5$ angles between the vectors as follows:

$$Q_1 = \sin^{-1}\left(\frac{\overrightarrow{AB_x}}{\sqrt{\overrightarrow{AB_x}^2 + \overrightarrow{AB_y}^2 + \overrightarrow{AB_z}^2}}\right) \quad (7)$$

$$Q_2 = \sin^{-1}\left(\frac{\overrightarrow{AB_y}}{\sqrt{\overrightarrow{AB_x}^2 + \overrightarrow{AB_y}^2 + \overrightarrow{AB_z}^2}}\right) \quad (8)$$

$$Q_3 = \sin^{-1}\left(\frac{\overrightarrow{BC_x}}{\sqrt{\overrightarrow{BC_x}^2 + \overrightarrow{BC_y}^2 + \overrightarrow{BC_z}^2}}\right) \quad (9)$$

$$Q_4 = \cos^{-1}\left(\underset{AC}{\rightarrow}^2 - \underset{AB}{\rightarrow}^2 - \underset{BC}{\rightarrow}^2\right) \quad (10)$$

$$Q_5 = \sin^{-1}\left(\frac{\overrightarrow{CD_x}}{\sqrt{\overrightarrow{CD_x}^2 + \overrightarrow{CD_y}^2 + \overrightarrow{CD_z}^2}}\right) \quad (11)$$

The angles values are then sent to the Robotic arms via a WLAN so as to move as the human standing in front of the kinect moving.

4. EXPERIMENTAL TEST RESULTS

Two practical cases are considered to test the mobile robot navigation at outdoor environment. In the first case, the robot navigates in an environment without obstacles in a cars park at Al-Nahrain University; the map of the searching is gotten from the Google earth pro program with resolution (919×1525pix) and area equal to (118×202 m^2). Buildings are detected and search area is divided into a grid with block size (1×1m^2), the starting location is received from the mobile robot GPS via the Arduino Wifi shield through a network and the goal location is selected by the user. A* algorithm is implemented and it creates a path between start and destination location. This path is sent to the mobile robot as a series of points in Geo formatting. The mobile robot doesn't reach its final destination (about 3 meters away) due to GPS inaccuracy. Depending on the feedback of a camera a manual control takes place to deliver it to the final destination. It was

successfully moved towards the target in about 10 to 20 centimeters away to enable the robotic arms to move freely. Figure 8 shows the selected path by the A* algorithm, the actual path of mobile robot with PID controller and the manual controlled path.

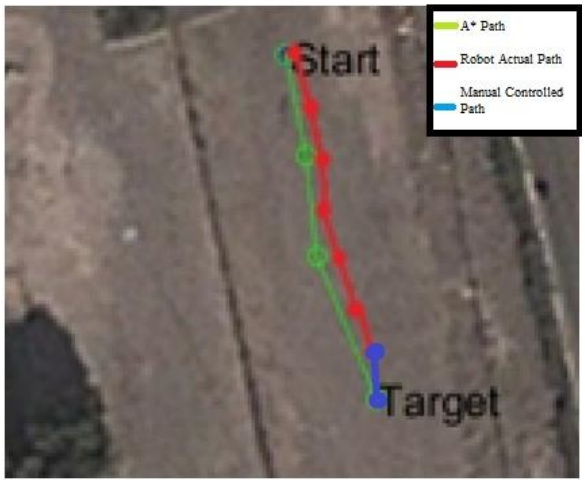


Figure 8: Mobile Robot Path

Figure 9 shows a snapshots from the camera before and after using the manual control to move the mobile robot towards the target. The left side of the figure the mobile robot is about 3 meters away from the target. After the manual control, the mobile robot is 15 centimeters from the target.

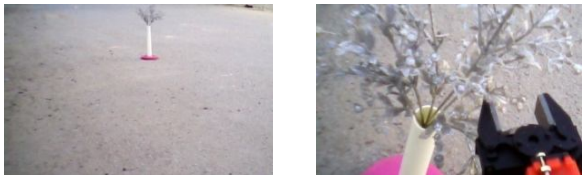


Figure: 9 Camera Snapshots before and after the manual control

Table 1 shows the recorded results for above test in addition to another test in a different place at Al-Nahrain University.

Table 1: Practical Results Recorded Of Mobile Robot Path

Path no.	Start location	Final target	Length of path (meter)	Slips from path (meter)	Error (meter)	Total time (min)
1	*Lat:* 33.280206 *Lng:* 44.377102	*Lat:* 33.279943 *Lng:* 44.377198	30.5	0.8	2.9	3.4
2	*Lat:* 33.278046 *Lng:* 44.374120	*Lat:* 33.278380 *Lng:* 44.374056	26.5	0.7	2.7	2.9

In table 1 it can be noticed that the mobile robot reaches the final destination by a miss distance less than three meters. This error is occurred due to GPS inaccuracy.

A Proportional-Integral-Derivative (PID) balanced speed controller is used for keep the robot moving straightly towards the next goal point in a specific angle. It depends on the reading of the digital compass to control the speed of the mobile robot motors, as shown in Fig.8 it is also enable the mobile robot to determine its path line to the target point with minimum error slips of less than 1 meter as shown in results table 1.

In the second test, the scenario of the first case is repeated but with local obstacles in the path of mobile robot. The mobile robot moves to the points of the path to reach the desired destination safely. Figure 10 shows how mobile robot behaves when there is more than one obstacle in its path.

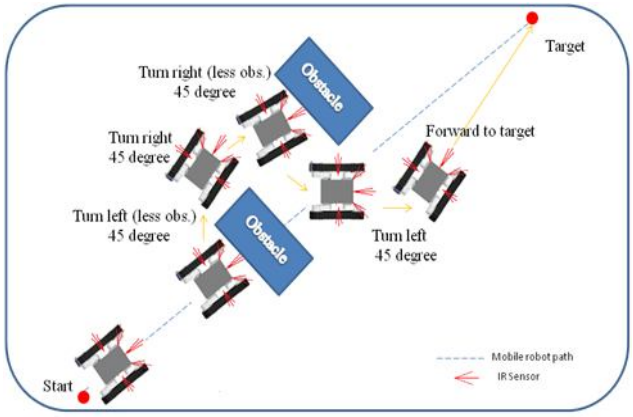
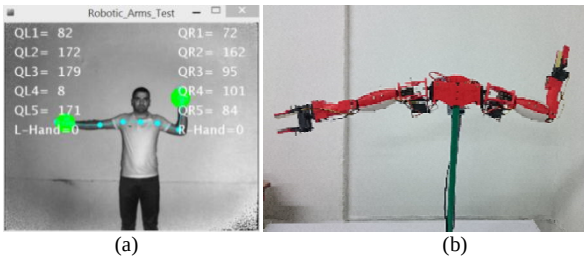


Figure 10: Mobile Robot behaviour with obstacles

The manual controlling is also used for moving the mobile robot at indoor environment. When the mobile robot reach its final destination, the Human Robotic Arms software application is started at robot from the base station in order to control the robotic arms. The camera at the robot acts as the human eyes that enable the user to interact with surrounding environment.

To control the humanoid robotic arms, an interface application at the PC base station was designed to allow the user to control the system as well as providing feedback to the user regarding their posture as shown in figure 11(a). Figure 11(b) shows the robotic arms response to human arms movement.



(a) (b)

Figure 11: Robotic Arms control

The server application at the robot should be run first to listen for a potential client. The client application should run immediately as follows: Connect to the server application using the server static IP address. Track user skeleton and

extract the X,Y, Z values each joint according following user movements. Inverse Kinematics then take place to calculate the required angle values. Kalman filter applied to the values of inverse kinematics before being sent to the robot. Figure 12 shows filtered data compared with raw data.

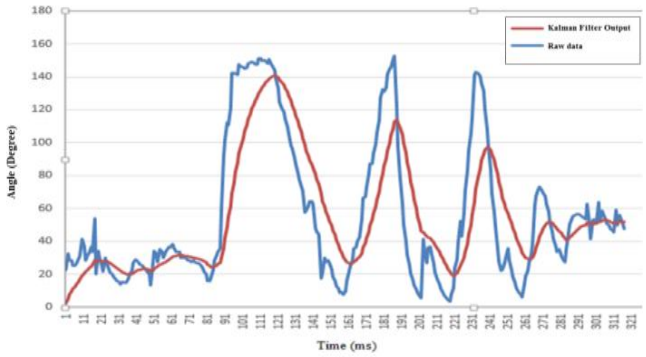


Figure 12: Filtered vs. Raw data

To test the validity and accuracy of the skeletal tracking algorithms carried out by Kinect sensor, three objects were chosen to be picked and held by the robotic arms: A bouquet of flowers, a bucket and a ball.

Due to the small size of the gripper, a small bouquet of flowers with a forked branches was chosen to be picked and held by the gripper. Figure 13 shows how the robotic arm picking, holding and dropping the bouquet of flowers from (location A) to (location B) which 3 meters away from (location A).

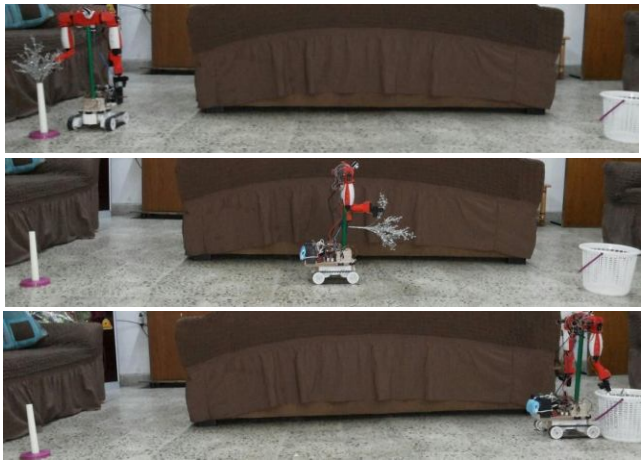


Figure 13: Picking, Holding and Dropping a small bouquet of flowers.

A bucket was picked and held by the robotic arm where the robotic arm used as hook cantered in the elbow joint. The bucket was transferred from (location A) to (location B) successfully as shown in figure 14.

Figure 14: Picking, Holding and Dropping a bucket.

The third test was to pick a ball using both robotic arms. The test failed to pick the ball because the tracking algorithm is corrupted when the user moves his arm in front of his body (main body) causing the tracked joint accuracy to be decreased, making the system to become instable in such case.

5. CONCLUSIONS

A* algorithm successfully finds the shortest path from satellite map of the search area that is divided into regular grid by using cell decomposition method.

The mobile robot successfully finds its path between its start location and target point, depending on GPS and digital compass information, it reaches nearly the target point with a miss distance of less than 3 meters due to GPS inaccuracy.

Depending on the feedback of the camera, a manual control from the PC base-station takes place to move the mobile robot to its final target.

The mobile robot successfully detects and avoids single and multiple obstacles in its path using five IR sensors.

It was proven that the 3D controlling of the HRA (using Microsoft Kinect sensor) does not need the utilization of any specialized devices which might attached to the user body in order to track the body movements.

The tracking algorithms of human skeleton appear to be running normally, but in some cases, defects have been arisen. When user moves his arm in front of any part of his body (e.g. head, main body, other arm, or legs) the tracked joint accuracy decreased, making the system to become unstable in such cases. This issue is considered as one of the major system weaknesses.

Wrist rolling is disabled when the hand is closed because the vector $\overrightarrow{CD}$ is disabled, since IK calculation depends on the vectors.

To complete the complex tasks, multi-Kinect will be used to work together, using a glove sensor (attached to the user hand) to calculate the wrist roll angle accurately, Using high quality servo motors with higher torque and accuracy and full size humanoid robotic arms in future work.

REFERENCES

1. V. J. Lumelsky, **Sensing, Intellegence, Motion, How Robots and Humans Move In Unstructured World,** John Wiley & Sons, Inc., Hoboken, New Jersey, 2006.
2. J. Gómez, A. F. Caballero, I. G. Varea, L. Rodriguez and C. R. Gonzalez**, A Taxonomy of Vision Systems for Ground Mobile Robots**, University of Castilla-La Mancha, Department of Computer Systems, Albacete, Spain, July 2014.
3. Kumar, K., Prasad, G., & Sreekanth, A., **Wireless Mobile Robotic Arm**, International Journal of Embedded & VSLI Systems, doi:10.1016/ j.proeng.2012.07.285, 2012.
4. J. M. Zogg, **GPS Basics,** u-blox ag, Switzerland, 2002.
5. Microsoft Corporation, **Meet Kinect for Windows**. Retrieved from https://dev.windows.com/en-us/kinect.
6. C. Veness. **Calculate distance, bearing and more between Latitude/Longitude points**. Retrieved from http://www.movabletype. co.uk/scripts/latlong.html.
7. M. Z. Al-Faiz and Ghufran E. Mahameda**. GPS-based Navigated Autonomous Robot**, International Journal of Emerging Trends in Engineering Research, Vol. 3, No. 4, April 2015.
8. M. Z. Al-Faiz and A. F. Shanta. **Humanoid Robotic Manipulator for Human**, Intelligent Control and Automation Vol. 6, No. 1, January 2015.