International Journal of Science and Advanced Technology

Volume 3 Issue 2

February, 2013

ISSN: 2221-8386 (Online)
2304-652X (Print)

Kinematic Analysis and Simulation of an Omnidirectional Mobile Robot

Ubaldo Geovanni Villaseñor Carrillo
Faculty of Informatics
Autonomous University of Queretaro
Queretaro, Mexico
ubaldo.villasenor@gmail.com

Efren Gorrostieta Hurtado
Faculty of Informatics
Autonomous University of Queretaro
Queretaro, Mexico
efrengorrostieta@gmail.com

Jose Emilio Vargas Soto
Faculty of Informatics
Autonomous University of Queretaro
Queretaro, Mexico
jsistdig@yahoo.com.mx

Saul Tovar Arriaga
Faculty of Informatics
Autonomous University of Queretaro
Queretaro, Mexico
caryoko@yahoo.com

Abstract—**In this paper, the simulation of an omnidirectional mobile robot is implemented. This paper will start with an explanation of the concept and advantages of omnidirectional mobility. Then, a brief introduction to the omnidirectional mobile robot will be given. Thereafter, the robot kinematic is analyzed and a motion simulation design is developed in C. Finally, experiments will be performed and results presented will be discussed.**

Keywords-Omnidirectional Robot; Kinematic; Open GL; Simulation.

I. INTRODUCTION

In the field of robotics, the importance of mobile robots is steadily increasing. Due to their freedom of movement, mobile robots are more flexible and can perform more tasks than their conventional fixed counterparts. A special class of mobile robots is omnidirectional robots. These robots are designed for 2D planar motion and are capable of translation (*x, y*) and rotation around their center of gravity (*θ*).

In the abstract of this paper, it was mentioned that the mobile robot, of which the movements are to be simulated, is capable of omnidirectional mobility. To gain a better understanding of this concept, it will now be discussed. Robotic vehicles are often designed for planar motion [1]. Some examples include floor cleaning or transport of goods in warehouses [2] [3]. In such a two-dimensional (*2D*) space, a body has three degrees of freedom (*3 DOF*) [4]. It can translate along the x and y axes and it can rotate around its center of gravity, the θ axis Fig.1.

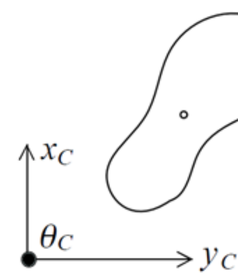


Figure 1. DOF (*x, y, θ*)

Most vehicles are not capable of controlling these three degrees of freedom independently, because of so called non holonomic constraints [5]. As example, consider a road where several cars are parked along the side of the road. If a driver wants to park his normal passenger car in an open space between two cars, he can't simply move sideways. The driver often has to drive forward and backward several times to make enough of an angle to insert his car into the free spot and to get a final orientation that is satisfactory. This is due to the inability of a car using skid steering to move perpendicular to its drive direction a non holonomic constraint. While generally such a vehicle can reach every location and orientation in a 2D space, it may require complicated maneuvers and complex path planning to do so, regardless of whether it is a human or robot controlled vehicle.

II. KINEMATIC MODELING

Now that a basic understanding of the omnidirectional robot has been developed, we can investigate the vehicle kinematics. If we want to prescribe the robot movements in the environment, we need to know how these variables relate to the primary variables we can control: the angular positions and velocities of the wheel shafts. Therefore, a kinematical model of the robot has to be developed. Fig.2. present the configuration of the robot, as well as all axis and relevant forces and velocities of the robotic system.

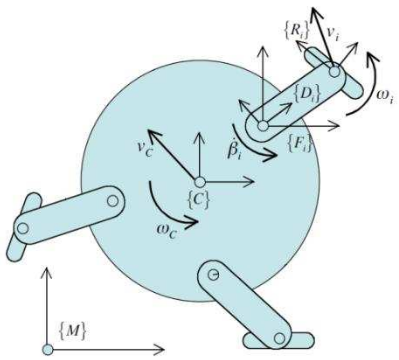


Figure 2. Placement of Coordinate Systems on a Wheeled Mobile Robot

The floor coordinate system (M) is stationary relative to the surface of travel and serves as the reference coordinate frame for robot motions. The robot coordinate system (C) is assigned to the robot body so that the position of the wheeled mobile robot is the displacement from the floor coordinate system to the robot coordinate system. The hip coordinate system (F_i) is assigned at the point on the robot body which intersects the steering axis of wheel (i). The steering coordinate system (D_i) is assigned at the same point along the steering axis of wheel (i), but is fixed relative to the steering link. We assign a contact point coordinate system (R_i) at the point of contact between each wheel and the floor. Coordinate system assignments are not unique. There is freedom to assign the coordinate systems at positions and orientations which lead to convenient structures of the kinematic model.

The coordinate system C is moving in three dimensions X, Y and θ. The coordinate systems $\bar{\mathrm{C}}$ and M are stationary; $\bar{\mathrm{C}}$ is an instantaneously, coincident coordinate system and M is a conventional reference coordinate system. The position of the moving coordinate system relative to its instantaneously coincident coordinate system is zero ($\bar{\mathrm{C}}_{\mathrm{PC}} = 0$). The position of the moving coordinate system relative the conventional reference coordinate system is non-zero ($\mathrm{M}_{\mathrm{PC}} \neq 0$). The non-zero velocity $\bar{\mathrm{C}}_{\mathrm{VC}}$ (acceleration $\bar{\mathrm{C}}_{\mathrm{aC}}$) of the moving coordinate system relative to the instantaneously coincident coordinate system is not equal to the velocity M_{VC} (acceleration M_{aC}) of the moving coordinate system relative to the conventional reference coordinate system. The velocity (acceleration) of the moving coordinate system relative to the conventional reference coordinate system M depends upon the position and orientation of the moving coordinate system relative to the reference coordinate system. The motivation for assigning instantaneously coincident coordinate systems is that the velocities (accelerations) of a multi-dimensional moving coordinate system can be computed or specified independently of the position of the moving coordinate system. The instantaneously coincident coordinate system is a conceptual tool which enables us to calculate the velocities and accelerations of a moving coordinate system relative to its instantaneous current position and orientation.

For stationary serial link manipulators, all joints are one dimensional lower pairs: prismatic joints allow Z motion and revolute joints allow θ motion. In contrast, wheeled mobile robot has three dimensional higher pair wheel to floor and robot to floor joints allowing simultaneous X, Y and θ motions. We assign an instantaneously coincident robot coordinate system $\bar{\mathrm{C}}$ at the same position and orientation in space as the robot coordinate system C. We define the instantaneously coincident robot coordinate system to be stationary relative to the floor coordinate system M. By design, the position and orientation of the robot coordinate system C and the instantaneously coincident robot coordinate system $\bar{\mathrm{C}}$ are identical, but (in general) the relative velocities and accelerations between the two coordinate systems are non-zero. When the robot coordinate system moves relative to the floor coordinate system, we assign a different instantaneously coincident coordinate system for each time instant. The instantaneously coincident robot coordinate system facilitates the specification of robot velocities (accelerations) independently of the robot position.

Homogeneous (4×4) transformation matrices are defined to express the relative positions and orientations of coordinate systems [4]. The homogeneous transformation matrix A_{Π_B} transforms the coordinates of the point B_r in coordinate frame B to its corresponding coordinates A_r in the coordinate frame A:

$$A_r = A_{\Pi_B}\, B_r \tag{1}$$

We adopt the following notation. Scalar quantities are denoted by lower case letters (ω). Vectors are denoted by lower case boldface letters (r). Matrices are denoted by upper case boldface letters (Π). Pre-superscripts denote reference coordinate systems. For example, A_r is the vector r in the A coordinate frame. The pre-superscript may be omitted if the coordinate frame is transparent from the context. Post-subscripts are used to denote coordinate systems or components of a vector or matrix. For example, the transformation matrix A_{Π_B} defines the position and orientation of coordinate system B relative to coordinate frame A; and r_x is the x component of the vector r.

Vectors denoting points in space, such as A_r in (1), consist of three cartesian coordinates and a scale factor as the fourth element.

$$A_r = \begin{pmatrix} A_{rx} \\ A_{ry} \\ A_{rz} \\ 1 \end{pmatrix} \tag{2}$$

We always use a scale factor of unity. Transformation matrices contain the (4×4) rotational matrix $(\mathrm{n\ o\ a})$, and the (3×1) translational vector.

$$A_{\Pi_B} = \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{3}$$

The three vector components n, o and a of the rotational matrix in (3) express the orientation of the x, y and z axes, respectively, of the B coordinate system relative to the A coordinate system and are thus orthonormal. The three components p_x , p_y and p_z of the translational vector p express the displacement of the origin of the B coordinate system relative to the origin of the A coordinate system along the x , y and z axes of the A coordinate system, respectively.

The aforementioned properties of a transformation matrix guarantee that its inverse always has the special form.

$$A_{\Pi_B^{-1}} = \begin{pmatrix} n_x & n_y & n_z & -(p.n) \\ o_x & o_y & o_z & -(p.o) \\ a_x & a_y & a_z & -(p.a) \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4}$$

Before we define the transformation matrices between the coordinate systems of our wheeled mobile robot model, we compile before our nomenclature for rotational and translational displacements, velocities and accelerations.

In general, any two coordinate systems A and B in our wheeled mobile robot model are located at non zero x, y and z coordinates relative to each other. The transformation matrix must therefore contain the translations $A_{d_{Bx}}$, $A_{d_{By}}$ and $A_{d_{Bz}}$. We have assigned all coordinate systems with the z axes perpendicular to the surface of travel, so that all rotations between coordinate systems are about the z axis. A transformation matrix in our wheeled mobile robot model thus embodies a rotation A_{θ_B} about the z axis of coordinate system A and the translations $A_{d_{Bx}}$, $A_{d_{By}}$ and $A_{d_{Bz}}$ along the respective coordinate axes.

$$A_{\Pi_B} = \begin{pmatrix} \cos A_{\theta_B} & -\sin A_{\theta_B} & 0 & A_{d_{Bx}} \\ \sin A_{\theta_B} & \cos A_{\theta_B} & 0 & A_{d_{By}} \\ 0 & 0 & 1 & A_{d_{Bz}} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5}$$

For zero rotational and translational displacements, the coordinate transformation matrix in (5) reduces to the identity matrix. We apply the inverse of the transformation matrix in (5) to calculate the position kinematics. By applying the inverse in (4) to the transformation matrix in (5).

In the interest to calculate the global position of the Wheeled Mobile Robot, we have to compute the forward kinematics. The kinematic model with respect to the robot coordinate system is given by (6) and (7).

$$p_i = \lambda_i + R(\alpha_i + \beta_i) * \delta_i \tag{6}$$

$$\theta_i = \alpha_i + \beta_i + \gamma_i \tag{7}$$

$$\begin{pmatrix} v_{cx} \\ v_{cy} \\ \omega_c \end{pmatrix} = \begin{pmatrix} c_i & -s_i & p_{iy} & -\lambda_{iy} \\ s_i & c_i & -p_{ix} & \lambda_{ix} \\ 0 & 0 & 1 & -1 \end{pmatrix} * \begin{pmatrix} v_{ix} \\ v_{iy} \\ \omega_i \\ \dot{\beta}_i \end{pmatrix} \tag{8}$$

$$V_c = \hat{J}_i * \hat{q}_i \tag{9}$$

Where p_i and θ_i are defined as the vector of position and orientation of the system R_i (system located in the contact point of the wheel with the ground) seen from C (Associated to the body of the robot, and is used as the vehicle guide point) and λ_i the position vector between C and the anchor point of the robot arm F_i.

Far as we can see, the speed of our robot depends of the speeds of the wheels, so the Jacobian calculation developed in the wheel J_i, is defined as the model that allows calculate the robot speed V_c.

As the linear speed of the wheel is obtained through the action of the motor (7), we get the following equation:

$$\hat{q}_i = \begin{pmatrix} 0 & 0 & 0 \\ -r_i & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} \omega_{ix} \\ \omega_{iy} \\ \dot{\beta}_i \end{pmatrix} \tag{10}$$

Where r_i is the radius of the wheel, ω_{ix} is the slewing speed and ω_i is the angular velocity of the wheel. Thus combining (6) and (8) we obtain the pattern guide point of the robot to calculate the velocity V_c.

$$J_i = \begin{pmatrix} r_i.s_i & p_{iy} & -\lambda_{iy} \\ -r_i.c_i & p_{ix} & \lambda_{ix} \\ 0 & 0 & -1 \end{pmatrix} \tag{11}$$

Because our robot has three wheels the system of equation is overdetermined, and this will give us the velocity vector V_c.

$$\begin{pmatrix} I \\ I \\ I \end{pmatrix}.V_c = \begin{pmatrix} J_1 & 0 & 0 \\ 0 & J_2 & 0 \\ 0 & 0 & J_3 \end{pmatrix}.\begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} \tag{12}$$

Using a least squares approximation is being sought a solution to the vector V_c.

$$A * V_c = B * q \tag{13}$$

$$V_c = (A^T * A)^{-1} * A^T * B * q \tag{14}$$

$$V_c = J * q \tag{15}$$

Where the matrix J represents the full Jacobian model robot.

$$v_{cx} = \frac{R(\omega_{2x} - \omega_{3x})}{\sqrt{3}} \quad (16)$$

$$v_{cy} = \frac{R(2\omega_{2x} - \omega_{2x} - \omega_{3x})}{3} \quad (17)$$

$$w_c = -\frac{R(\omega_{1x} - \omega_{2x} - \omega_{3x})}{3L} \quad (18)$$

By placing the result in matrix form we get the mobile robot Jacobian function.

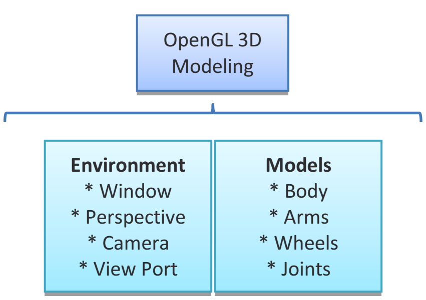
$$\begin{pmatrix} v_x \\ v_y \\ \omega_c \end{pmatrix} = \begin{pmatrix} 0 & \frac{R}{\sqrt{3}} & \frac{R}{\sqrt{3}} \\ \frac{2R}{3} & \frac{-R}{3} & \frac{-R}{3} \\ \frac{-R}{3L} & \frac{-R}{3L} & \frac{-R}{3L} \end{pmatrix} \cdot \begin{pmatrix} \omega_{1x} \\ \omega_{2x} \\ \omega_{3x} \end{pmatrix} \quad (19)$$

]

III. SIMULATION

One of the main objectives of this paper is to illustrate various features in OpenGL which will demonstrate model rendering, environment setting and interactive control capabilities. Another objective of this simulation program is to avoid any miss behavior and collision of the robot in the physical world control. Therefore, this paper implements a forward kinematics analysis method to calculate the final position of the wheels.

OpenGL is the industry most widely used, supported and best documented 2D/3D graphics API; this makes it inexpensive and easy to obtain information on implementing computer graphics. OpenGL is supported and complete independence on most operating system and it is compatible with most programming languages. All OpenGL applications produce consistent visual display results on any OpenGL API compliant hardware, regardless of operating system or windows system [6]. However, the basic OpenGL library is not capable of opening windows or responding to interrupts from a mouse or keyboard [7]. Fig.3. illustrates the 3D model rendering structure of the Wheeled Mobile Robot in the simulation program which consists of four modules; model, environment, visual effect and user control.



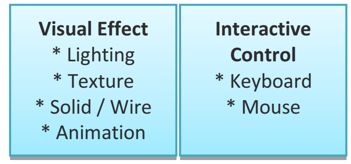


Figure 3. Model Rendering Structure of the Wheeled Mobile Robot

A. *Model Rendering*

In order to construct the Wheeled Mobile Robot, this program uses a Virtual Reality Modeling Language files from SolidWorks to make the simulation more realistic as shown in Fig.4. and Fig.5.

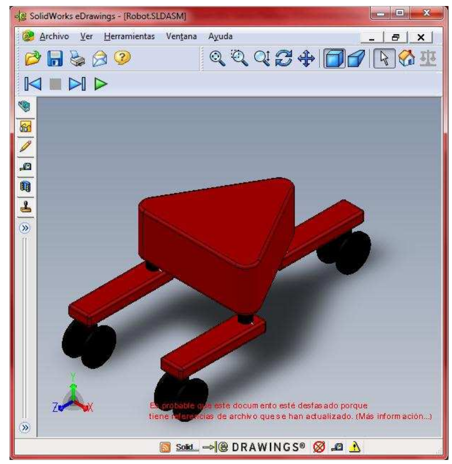


Figure 4. SolidWorks of the Wheeled Mobile Robot

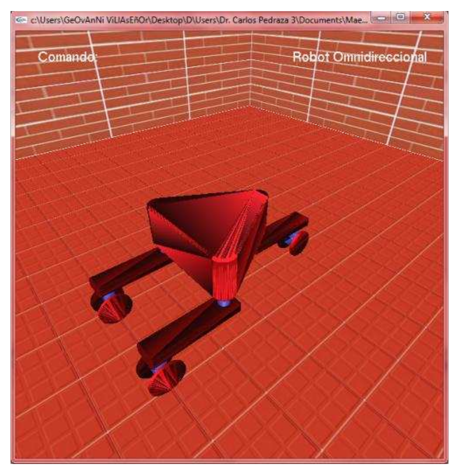


Figure 5. OpenGL Objects of the Wheeled Mobile Robot

B. *Interactive Control*

The robot joints and the visual effect that implemented in this program are controlled by the keyboard and mouse; it can support callback functions which are provided by GLUT library. The program read number keys from 0 to 9 and letter keys for a to z to control different joints of the robot, camara and scale effects. It also explains other shortcut key to switch between different camara views and the scale effect; and one more shortcut to run the kinematic algorithm of the Wheeled Mobile Robot. Other than keyboard control, this program allows user to use the right click from mouse input to accommodate the view camara of the user control as shown as Fig.6-9.

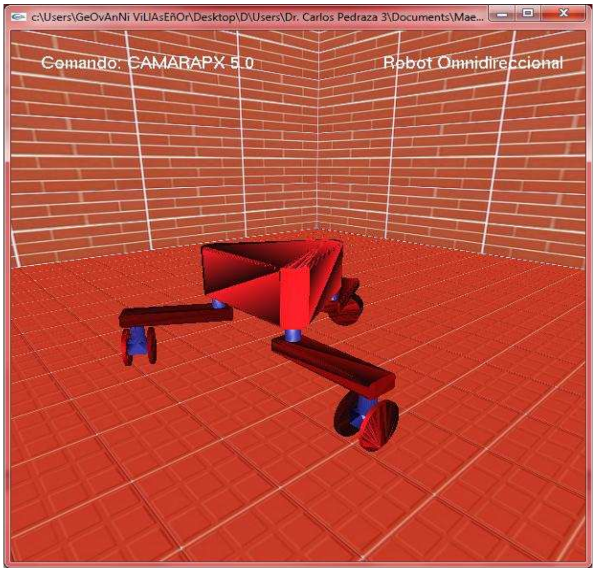


Figure 6. OpenGL Visual Camara Effect of the Wheeled Mobile Robot

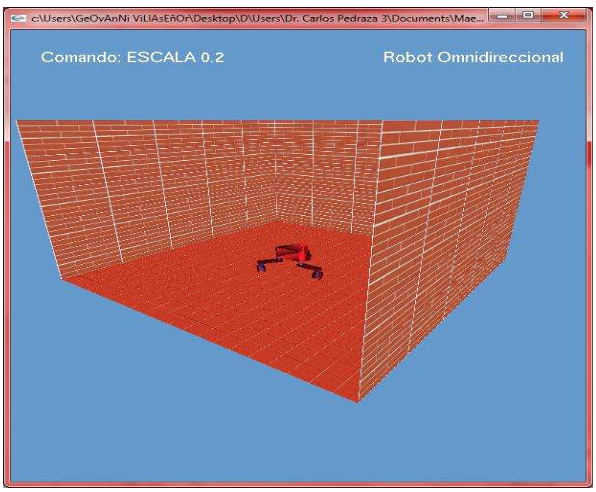


Figure 7. OpenGL Visual Scale Effect of the Wheeled Mobile Robot

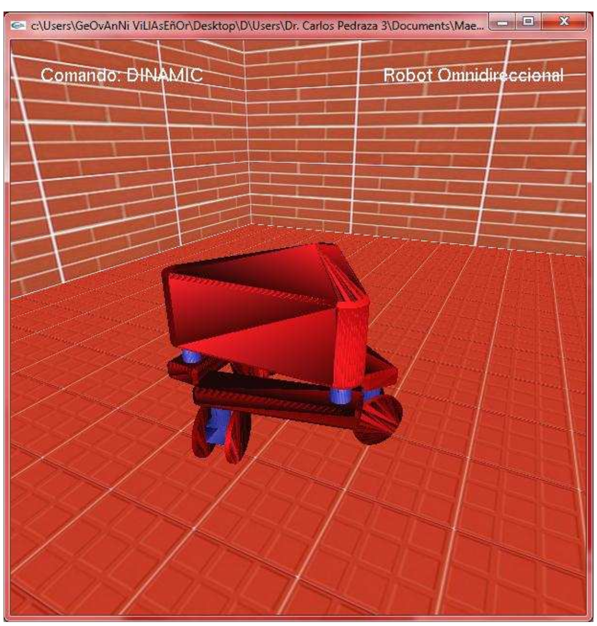
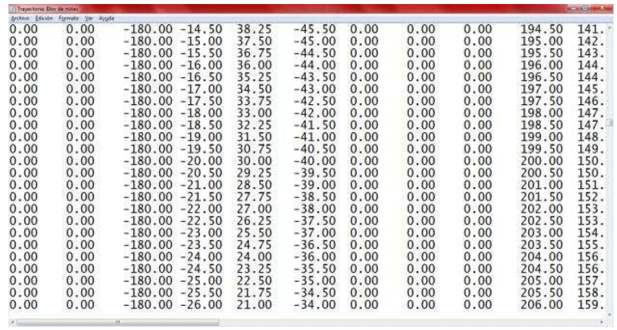


Figure 8. OpenGL Dinamic Simulation of the Wheeled Mobile Robot



0.00	0.00	-180.00	-14.50	38.25	-45.50	0.00	0.00	0.00	194.50	141.
0.00	0.00	-180.00	-15.00	37.50	-45.00	0.00	0.00	0.00	195.00	142.
0.00	0.00	-180.00	-15.50	36.75	-44.50	0.00	0.00	0.00	195.50	143.
0.00	0.00	-180.00	-16.00	36.00	-44.00	0.00	0.00	0.00	196.00	144.
0.00	0.00	-180.00	-16.50	35.25	-43.50	0.00	0.00	0.00	196.50	144.
0.00	0.00	-180.00	-17.00	34.50	-43.00	0.00	0.00	0.00	197.00	145.
0.00	0.00	-180.00	-17.50	33.75	-42.50	0.00	0.00	0.00	197.50	146.
0.00	0.00	-180.00	-18.00	33.00	-42.00	0.00	0.00	0.00	198.00	147.
0.00	0.00	-180.00	-18.50	32.25	-41.50	0.00	0.00	0.00	198.50	147.
0.00	0.00	-180.00	-19.00	31.50	-41.00	0.00	0.00	0.00	199.00	148.
0.00	0.00	-180.00	-19.50	30.75	-40.50	0.00	0.00	0.00	199.50	149.
0.00	0.00	-180.00	-20.00	30.00	-40.00	0.00	0.00	0.00	200.00	150.
0.00	0.00	-180.00	-20.50	29.25	-39.50	0.00	0.00	0.00	200.50	150.
0.00	0.00	-180.00	-21.00	28.50	-39.00	0.00	0.00	0.00	201.00	151.
0.00	0.00	-180.00	-21.50	27.75	-38.50	0.00	0.00	0.00	201.50	152.
0.00	0.00	-180.00	-22.00	27.00	-38.00	0.00	0.00	0.00	202.00	153.
0.00	0.00	-180.00	-22.50	26.25	-37.50	0.00	0.00	0.00	202.50	153.
0.00	0.00	-180.00	-23.00	25.50	-37.00	0.00	0.00	0.00	203.00	154.
0.00	0.00	-180.00	-23.50	24.75	-36.50	0.00	0.00	0.00	203.50	155.
0.00	0.00	-180.00	-24.00	24.00	-36.00	0.00	0.00	0.00	204.00	156.
0.00	0.00	-180.00	-24.50	23.25	-35.50	0.00	0.00	0.00	204.50	156.
0.00	0.00	-180.00	-25.00	22.50	-35.00	0.00	0.00	0.00	205.00	157.
0.00	0.00	-180.00	-25.50	21.75	-34.50	0.00	0.00	0.00	205.50	158.
0.00	0.00	-180.00	-26.00	21.00	-34.00	0.00	0.00	0.00	206.00	159.

Figure 9. Result Angles and Positions of the Wheeled Mobile Robot

IV. RESULTS

In this section for verification of the presented system, the model is implemented in the Wheeled Mobile Robot Fig.10-12.

Figure 10. Wheeled Mobile Robot (Contracted)

Figure 11. Wheeled Mobile Robot (Extended)

Figure 12. Wheeled Mobile Robot (Initial Position)

The initial values of the wheeled mobile robot can be seen in Table 1.

	Wheel 1	Wheel 2	Wheel 3
α	90	270	270
β	45	45	45
γ	270	270	90
δ	(-223.66, -122.85)	(99.71, -23.14)	(24.75,106.7)
λ	(-61.86, 56.72)	(-136.86 -73.19)	(13.14, -73.19)

Table 1. Initial Values

And a general view of kinematic results (Slewing Velocity, Angular Velocities) of the entire Wheeled Mobile Robot can be seen in Table 2.

Wheel 1				
Slewing Velocity	0.55	rps	355.44	mm/s
Angular Velocity	3.45	rad/s	- 804.22	mm/s
Angular Velocity (Arm)	3.45	rad/s	0	rad/s
Wheel 2				
Slewing Velocity	0.55	rps	203.98	mm/s
Angular Velocity	3.45	rad/s	-292.63	mm/s
Angular Velocity (Arm)	3.45	rad/s	0	rad/s
Wheel 3				
Slewing Velocity	0.55	rps	312.29	mm/s
Angular Velocity	3.45	rad/s	-124.92	mm/s
Angular Velocity (Arm)	3.45	rad/s	0	rad/s

Table 2. Kinematic Results

V. Conclusion and Future Work

This research has presented a kinematic system that is valid for one type of Wheeled Mobile Robot. The kinematic simulation program only runs as a standalone application. This simulation has all the behavior data of the wheeled mobile robot; the reason is that we will integrate it to real motion controller program. We will include the collision detection system, it will be extended to the wheels and consider the intersection between an obstacle and the robot. Other visual effect enhancements like material, shadow and reflection will be implemented to make the model more realistic and the dynamic model.

References

[1] Fish, S., 2004, "UGV's in Future Combat Systems" Proceedings of the SPIE – The International Society for Optical Engineering, Unmanned Ground Vehicle Technology VI, Vol. 5422, pp. 288 - 291.

[2] Blitch, J., 1996, "Artificial Intelligence Technologies for Robot Assisted Urban Search and Rescue" Expert Systems with Applications, 11 (2), pp. 109 – 124.

[3] Erickson, J., 2006, "Living the Dream: An Overview of the Mars Exploration Project" IEEE Trans. Rob. Autom., 13 (2), pp. 12 – 18.

[4] Cheng, J., Gao, L., and Wang, H., 2006, "Steering Analysis of Tracked Vehicles Based on Skid Condition" Chin. J. Mech. Eng., 42, pp. 192 – 195.

[5] Park, T., Lee, J., Yi, B., Kim, W., and You, B., Oh, S., 2002, "Optimal Design and Actuator Sizing of Redundantly Actuated Omni-Directional Mobile Robots" Proceedings of the IEEE International Conference on Robotics and Automation, pp. 732 – 737.

[6] OpenGL, OpenGL API Documentation http://www.opengl.org/documentation/, 2010.

[7] Whitrow, R., 2008, "OpenGL Graphics Through Applications" Springer-Verlag London Limited, pp. 145 – 153.

Ubaldo Geovanni Villaseñor Carrillo was born in Mexico City, Mexico, in 1988. He received the B.S. degree from Mexico Valley University, Queretaro, Mexico in 2010. His current research interests include the Robotic Design and Embedded Systems.