Proceedings of International Seminar on Applied Technology, Science, and Arts (2nd APTECS), Surabaya, 21-22 Dec. 2010, ISSN 2086-1931

Robot Soccer System Based on Virtual Force Field Method Approach

RIZKY YUNIAR HAKKUN[1], ENDAH SURYAWATI NINGRUM[1], AND SETIAWARDHANA[1]

[1]Politeknik Elektronika Negeri Surabaya,
Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia
email: {rizky, endah, setia}@eepis-its.edu

Abstract— **This paper discusses the methods used to determine the robot path and avoid obstacles. The method used is based on developing the Virtual Force Field method.From the robot position obtained from the camera, that is calculated each resultant of obstructions. Furthermore, the robot will move towards the target of the ball. But if there is an obstacle in front of the robot will move toward the direction of the resultant. Output sent to the robot via a serial module and the subsequent results visualize the movement of the robot in the simulation program.**

Keywords— **soccer robot, virtual force field, vector**

1. Introduction

The issues discussed are as follows :

- The parameters used in the process of determining the object and the target.
- The parameters used as input to the module, which is used to determine the paths and avoid collisions.
- How to convert the input data of the position of the robot and other objects from the camera into a variable so that it can obtain input and output parameters.
- How to visualize the results of a structured program in a simulation program.

As for the limitations problem created is as follows:

- This study assumes the field size used was 2.2 m x 1.8 m
- Just make a rule of robot soccer game.
- Terrain is considered flat trajectory
- The number of robots is 3.
- Visualization presented in 2D animation.
- Connection for data transmission using the serial port.

2. Related Works

Various studies have been conducted since 1991 and has gained a lot of varying results. One of them is on the robot Caramel. In this study, the sensor used is an ultrasonic sensor. This robot is designed to have some ability to fix the limitations and weaknesses in Potential Field primitive methods, such as the ability to define path, avoid collisions, able to move quickly and based on real-time.

In this paper, camera will use as a sensor. The camera serves to acquire data, such as the position of objects contained in the field as a input [3]. Then the method was applied to the PC, and the results will be sent to the robot via a wireless module [3]. In this paper does not use 2D histogram grid as used by Koren, Y. and Borenstein, J [2] but the calculation is based on the vector and Cartesian coordinates (x, y) so that the desired results more details .

3. Methodology

The research methodology can be seen in the following figure. The scope of this research is on the inside of the red box.

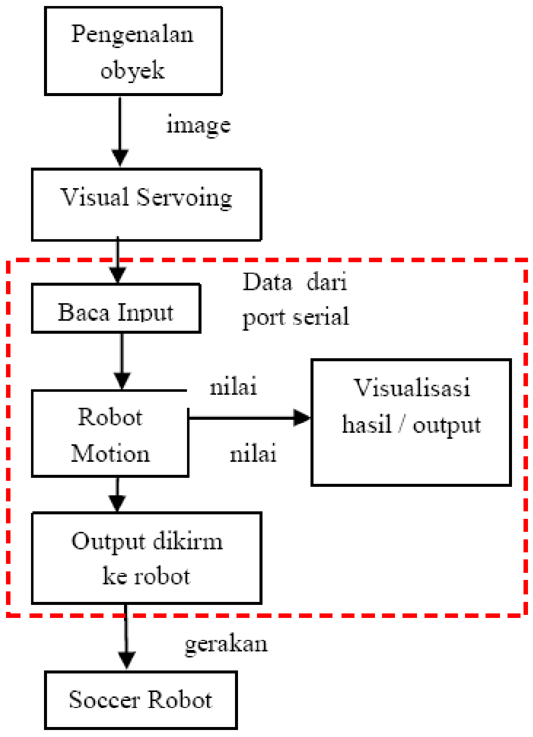


Figure 1. Design System

3.1 Input from Sensor

After doing a visual servoing system and acquiring data in the field, then the output will be sent via the serial port. Visual servoing is object detection using a camera. Data is sent in the form of object position data (x, y). Because the visual servoing system and the simulation program are in a different PC then the delivery done via the serial port.Input in the form:

- current robot position
- ball / target position
- the position of the robot opponent

After input is received then the program will do the parsing of data to obtain the value of x and y. Parsing data is done to get the value of x and y values In the delivery of data sent with the following format :

“$ax_1y_1x_2y_2x_3y_3$”

With each of the 3 digits for values of x and y are sent, with a character as a header.Header to use to mark that a data will be entered. If a character has a header in addition to the incoming data after the character will not be stored. If a data has a header then it will ensure that subsequent data is stored in an array.

3.2 Motion Planning

3.2.1 Game Rule

- Rule game is a rule that should be used as a benchmark of the robot movement. This rule will be the basis for determining the game strategy. Rule game on this system is:
- Robots have the ball toward the target, and avoid the opponent robots
- Opponent moves so that the system used is based on real time
- The robot has a range with a specific value with the central point is the midpoint of the robot.
- First robot will determine how much distance the target and obstacle against him. If the distance of an object is less than the range of the robot must avoid the obstacle in a certain direction.
- From the known position that is calculated as the resultant of vectors robots, opponents and targets. Then you will know the resultant direction.
- Calculation of the resultant done continuously along with the position of a moving robot and the opponent (real time).
- Output of the program are sent to the robot via serial port.
- The process stops when the robot has reached the target position.

3.2.2 Motion Planning

Motion Planning includes path planning and collision avoidance. Determination of paths used to determine the position of other robots, the position of the object and the target (in this case the ball and the wicket) and determine the path that must be passed by the robot from initial position to goal position that has been determined by criteria that route is best. While collision avoidance is used when the robot finds the position of obstacle at a certain distance and then move to avoid it. There are three layers in this phase :

(1) Vector Formulation

Formulation of the vector used to calculate the vector of each object, the robot, obstacle and the target. The steps are :

1. Calculate the vector of each object from a known position (x, y)

$$F = \sqrt{x^2 + y^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

2. From Known position, calculate the object angle of the x-axis

$$\tan \alpha = \frac{y}{x}$$

$$\alpha = a\tan \frac{y}{x} \quad \ldots\ldots\ldots\ldots\ldots\ldots (2)$$

There are α1, α2, α3 because there are three objects in the field of robot, obstacle and the robot.

3. Now we know the angle of each, then calculate the difference of angle between two vectors :

$\Theta = \alpha_1 - \alpha_2$

For $\alpha_1 < \alpha_2$ and otherwise.

4. With cosines formula, calculate the resultant robot one with another object

$$R = \sqrt{F_1^{\,2} + F_2^{\,2} + 2F_1F_2 \cos\angle(F_1, F_2)} \quad \ldots (3)$$

5. Determining the direction of the resultant

$$\sin \sigma = \frac{F_1 * \sin(180^o - \theta)}{R} \quad \ldots\ldots\ldots\ldots (4)$$

(2) Obstacle Mapping

Mapping obstacle performed to detect the presence of obstacle around the robot. The first time calculated the distance between the robot with the obstacle using Euclidian distance formula.

In this simulation program has been determined that the range value 40. Of 40 this means that the robot has a range / range as far as 40 pixels from the center point.

(3) Obstacle Avoidence layer

In this layer to be checked against the obstacle the robot position, whether the obstacle is in range or out of range. If the obstacle is in the range of the robot will move away and if not then the robot will move toward the target. Terms of collision avoidance is done by calculating obstacle in getting directions and then the resultant inter-robot with obstacle. In the simulation program, calculation of Euclidian distance performed from the obstacle to robot. If the distance is less than the range (in this case the range specified range = 40) then the robot moves in the direction of the resultant avoid the obstacle. And if the distance exceeds the range of the robot moving towards the target.

3.3 Output Delivery

Output is sent is the position of the target and the size of the resultant corner. Delivery of output is done using the serial port.

3.4 Simulation Design

The design of the simulation program using the JDK (Java Development Kit) 6.6 Netbeans 6.8 with additional libraries and the Java Game Engine GTGE javax.comm package for access to the serial port. There are several classes :

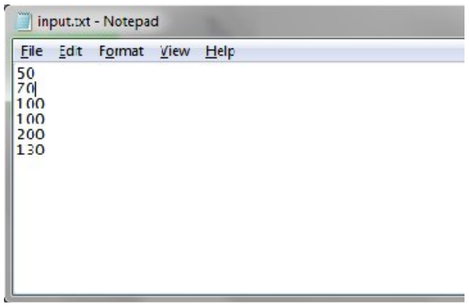
- Simulation.java Class This class is the parent of all classes and will be called the first time when the program runs. This class is made to extend the class to GameEngine GTGE. Game Engine is a set of all objects in a program like the look (graphics), sound, etc.
- MenuParent.java Class This class is used to initialize the initial values of variables such as initial position of the object. Also used to read input from serial port.
- SimpleSerialJava Class This class is used to access the serial port.
- Inputan.java Class This class is used to display beginning of the program and used to call a function of the input data. In this class there are buttons that when clicked it will call the class Animation.java.
- Animation.java Class This class is used to visualize simulation program that contains the movement of robots on the field. After receiving initial input from the input of the class will be called this class. This class includes 3 layer above: namely the rule of games, mapping obstacle and collision avoidance.
- Vektor.java Class This class contains methods used to calculate the vector, such as calculation of angles and calculation of the direction / angle the resultant outcome.This class is always called when the program took the decision in determining the resultant.

4. Experimental Result

Software testing done in 2 ways Testing Without Integration with Serial Input (only a simulation program) and testing the integration with serial input.

4.1 Integration Testing Without the serial input (simulation program)

Testing without integration with a serial input which means it is software testing performed without taking input from the serial port. Input is taken from the text data, which are read by the program.
This is data that is stored in the format .txt:



```
50
70
100
100
200
130
```

Figure 4. 1 Data input

Each line represents the value of x and y for each object. The program will perform readings from the first row until last and then store them in array.

Data is stored in an array variable tamping []. This function is in class MenuParent.Then these variables will in subsitusikan on a variable that holds the position of each object

4.2 Single Player Game

Single player game consists of a robot, an opponent and 1 ball. In the single player game is a robot given the initial position of example (as in the input text):

Robot 1 : (50,70)
Opponent Robot: (100,100)
Ball : (200,130)

Of the three positions above can be described as follows :

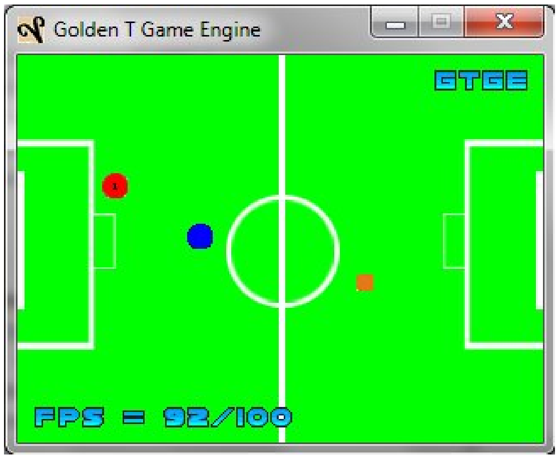


Figure 4. 2 The initial position of robots, the opponent and the ball

Black box is the range / range of mapping obstacle called the active window. Range is calculated from:

```
double range = pos_robot_x + 100;
```

This Active window will continue to follow the movement of the robot moves in invisible (not visible on screen). If a target / ball is in the active window, the program will calculate the resultant between the robot opponent with the ball. And if the target / ball is outside the active window, the program will calculate the resultant between the wall with the robot opponent.

When moving, the program will calculate:

❖ Calculate vector between the robot with a big opponent by using the formula of Pythagoras, we can get besarya displacement vector between the robot with the opponent. Because the robot as the central point (0.0) then can be obtained:

$$F_r = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

With x2 = x position of obstacle, x1 = x position of the robot, y2 = y position of obstacle and y1 is y position of robot. So the position can be obtained diata :

$$F_r = \sqrt{(100-50)^2 + (100-70)^2}$$
$$= \sqrt{50^2 + 30^2}$$
$$= \sqrt{2500 + 900}$$
$$= \sqrt{3400}$$
$$= 58{,}3095$$

- Calculate the vector of the robot with a target by using the same formula it can be obtained :

$$F_t = \sqrt{(200-50)^2 + (130-70)^2}$$
$$= \sqrt{150^2 + 60^2}$$
$$= \sqrt{22500 + 3600}$$
$$= \sqrt{26100}$$
$$= 161{,}5549$$

Calculating the angle between the robot with the opposite of the opposite corner of the robot is obtained from:

$$\tan\alpha = \frac{y}{x}$$

With y is a (position versus y - y position of the robot) and x is the (opponent's position x - x position of the robot). By using the formula above, the angle between the robot with the opponent is:

$$\tan\alpha = \frac{y}{x}$$
$$= \frac{(pos\_obs\_y - pos\_robot\_y)}{(pos\_obs\_x - pos\_robot\_x)}$$
$$= \frac{100-70}{100-50}$$
$$= \frac{30}{50}$$
$$= 0{,}6$$
$$\alpha = a\tan 0{,}6$$
$$= 21.801409486351815$$
$$= 21.8$$

- Calculating the angle between the robot with the target. To calculate the angle between the robot with the target using the same manner as above. It's just for x2 and y2 are each target position x and y position of the target. By using the formula above, the angle between the robot with the target is:

$$\tan\alpha = \frac{y}{x}$$
$$= \frac{(pos\_bola\_y - pos\_robot\_y)}{(pos\_bola\_x - pos\_robot\_x)}$$
$$= \frac{130-70}{200-50}$$
$$= \frac{60}{150}$$
$$= 0{,}4$$
$$\alpha = a\tan 0.4$$
$$= 30.963756532073525$$
$$= 30.964$$

So the angles are flanked by a robot, opponent and the target is:

21,8+30,964 = 52,764

- Calculate the magnitude and direction of the resultant. To calculate the amount of resultant formula can be used :

$$R = \sqrt{F_1{}^2 + F_2{}^2 + 2 * F_1 * F_2 * \cos\alpha}$$

By using the formula above with F1=58,3095 and F2 = 161,5549 and α = 52,764, then :

$$R = \sqrt{F_1^2 + F_2^2 + 2 * F_1 * F_2 * \cos\alpha}$$
$$= \sqrt{(58{,}3095)^2 + (161{,}5549)^2 + 2 * 58{,}3095 * 161{,}5549 * \cos(52{,}764)}$$
$$= \sqrt{3400 + 26100 + 11400{,}2}$$
$$= \sqrt{40900{,}3}$$
$$= 202{,}24$$

To calculate the angles in the resultant use the following formula:

$$\sin\sigma = \frac{F_1 * \sin(180^\circ - \theta)}{R}$$

With Θ is the wedge angle between the robot, opponent and target. By using the formula above, the resultant angle is large :

$$\sin\sigma = \frac{161{,}5549 * \sin(180-52{,}764)}{202{,}24}$$
$$= \frac{161{,}5549 * \sin(127{,}236)}{202{,}24}$$
$$= \frac{161{,}5549 * 0{,}79}{202{,}24}$$
$$= \frac{128{,}622}{202{,}24}$$
$$= 0{,}6359$$

4.3 The testing carried out by integration with serial input

Testing I :

Before performing the test on the software, test the connection between 2 PCs using hyperterminal. For example we make a connection on port COM1.

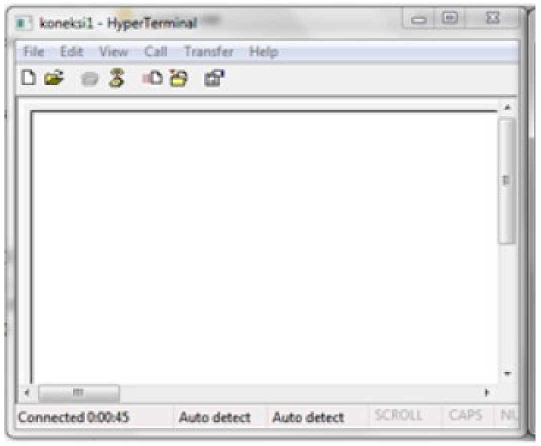


Figure 4. Data sent from COM1 toCOM2

Data obtained from serial port input as follows:

Robot : (108, 134)

- Opponent Robot : (167, 104)
- Target : (274, 88)

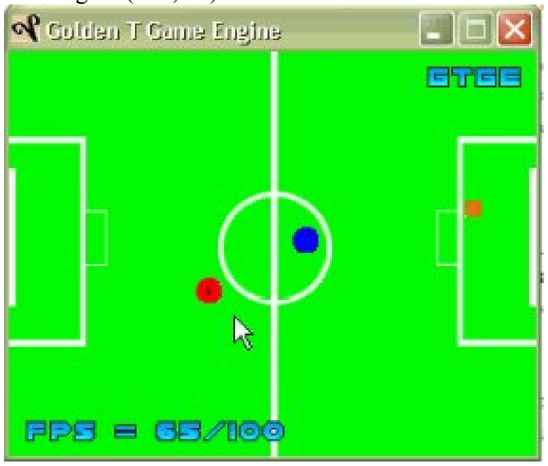


Figure 5. Simulation program

After input from serial port into the robot, target and your opponent will be positioned at the starting position obtained from the serial input.

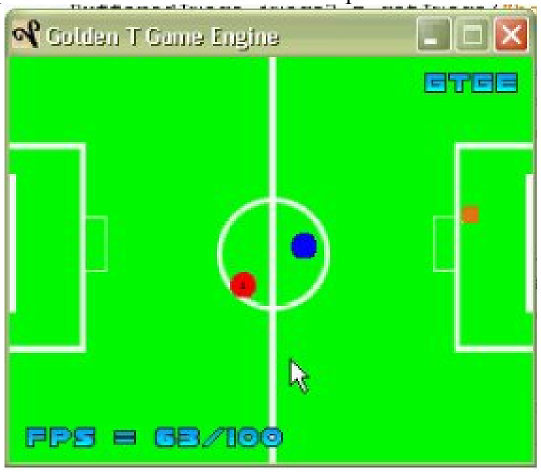


Figure 6. Robot moving

The robot starts moving towards the target. In this experiment did not use a real robot but use a marker which is positioned as a robot, target and obstacle. The robot is driven manually by hand. Then by using this simulation program can be seen the movement generated by the VFF method that makes the robot can avoid obstacles.And here are the results:

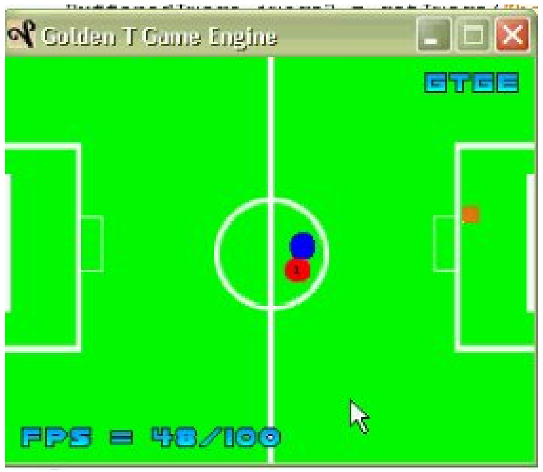


Figure 7. Robot hit an obstacle because the position was not updated

The figure above shows that the robot hit a snag. Because the marker is not updated so that the data used is data starting position and it caused the robot hit an obstacle.

Testing II :

This experiment is the same as experiment I but the robot is driven with a manual.

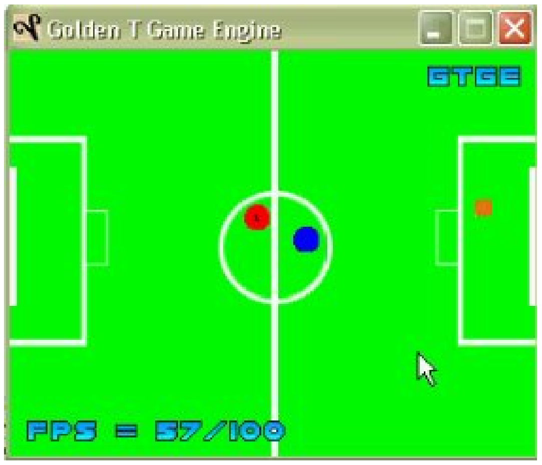


Figure 8. Robot succed to avoid obstacle

In this second experiment the position of the robot, obstacle and the target is updated manually. Manual that is meant here is the marker is moved manually as a robot.

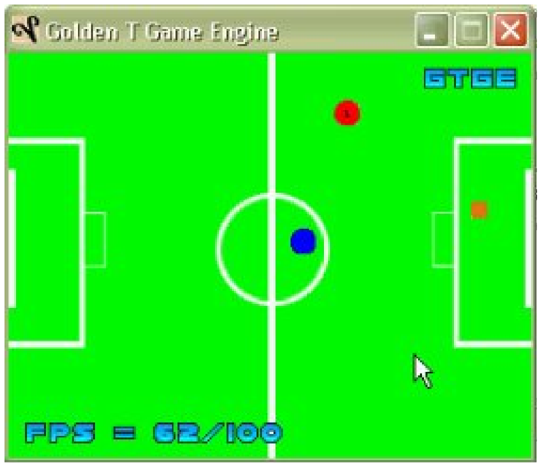


Figure 9. Robot toward target

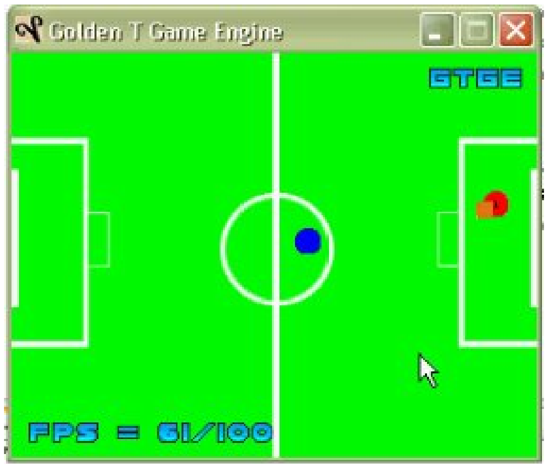


Figure 10. Robot successfully reach the target

5. Conclusion

Conclusion

- The closer the robot to the obstacle then thrust a virtual robot to the obstacle the greater and vice versa.
- This application is able to work in real time because it is able to update the position even though the execution has not completed the previous movement.
- This application can be developed as needed for example to be developed at the game with single player or multiplayer modes.
- This application is capable of parsing data quickly and accurately so that there is no guarantee that the received data input is valid data.
- VFF method proved to have a better performance than the conventional method of being able to work in real time.
- The weakness of this system is that if the obstacle / barrier complex shaped like a hitch in the form of a barrier wall.

Suggestion :

- Robot motion on this application still look stiff and need an improvement to the display / visualization.
- In further developments, it is expected that the application was made to have the facility to synchronize the speed of data between the receiver and sender of data.Because if the data is sent too late to influence decision making in the movement of the robot.
- In the further development of this program is expected to be applied to the actual robot, so no need to use markers.

6. References

[1] http://www.robotstorehk.com/soccer/soccer.html (4 September 2009)

[2] Supriadi, Ahmad. Perencanaan Jalur dan Penghindaran Tabarakan pada Robot POEMAV. Departemen Teknik Elektro, Institut Teknologi Bandung. Desember 2005. url : http://s.itb.ac.id/%7Enarpen/kuliah/ta1/contoh%20makalah%20TA%201/Paper%20Seminar%20ahmad.doc (diakses 22 Nopember 2009)

[3] Koren,Y. Senior Member, IEEE and Borenstein, J., Member, IEEE The University of Michigan, Ann Arbor. Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation. IEEE Conference on Robotics and Automation, Sacramento California, pp 1398-1404. April 1991. Url :

[4] http://wwwpersonal.umich.edu/~johannb/Papers/paper27.pdf .(diakses : 8 September 2009).

[5] Latombe, Jean-Claude. Robot Motion Planning. Kluwer Academic Publishers. Massachusetts. 1991.

[6] Bastan, Muhammet. Visual Servoing of Mobile Rrobots Using Potential Fields.2004. (diakses : 15 Oktober 2009).

[7] Prahlad Vadakkepat, Tong Heng Lee and Liu Xin. Application of Evolutionary Artificial Potential Field in Robot Soccer System. National University of Singapore, Singapore. 2001. url : http://www.ece.nus.edu.sg/stfpage/elepv/publication/APF_RSS.PDF.(diakses 8 September 2009).

[8] J.C. Wolf , P. Robinson, J.M. Davies. Vector Field Path Planning and Control of An Autonomous Robot In a Dynamic Environment. url : http://www.swrtec.de/swrtec/research/publications/VECTOR_FIELD_PATH_PLANNING_AND_CONTROL_OF_AN_AUTONOMOUS_ROBOT_IN_A_DYNAMIC_ENVIRONMENT.pdf (diakses 12 Desember 2009)

[9] J. Borenstein, Y. Koren. The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots. IEEE Journal of Robotics and Automation Vol 7, No 3, , pp. 278-288. June 1991. url : http://eprints.kfupm.edu.sa/71681/1/71681.pdf (diakses 12 Desember 2009)

[10] http://wss-id.org/blogs/nikita_manezz_sby/archive/2007/09/24/robot-sepak-bola.aspx

[11] http://www.gurumuda.com/fisika-sma/Penjumlahan%20vektor.pdf

[12] www.tofi.or.id/download_file/Kul_1_VEKTOR.ppt

[13] http://id.wikipedia.org/wiki/Vektor_(spasial)

[14] http://www.goldenstudios.or.id/products/GTGE/index.php

[15] http://dhimaskasep.files.wordpress.com/2008/02/osp06-komunikasi-data-pada-sistem-manufaktur.pdf

[16] http://p_musa.staff.gunadarma.ac.id/Downloads/files/5117/lecKK-012325-5-1.pdf