Control and Analysis of an Omnidirectional Autonomous Robot Based on Software Approach and Multimedia Database

Mohsen Taheri*, Mohammad Naderi Dehkordi, Mehran Sharafi**, Mohammadhossein Nadimi-Shahraki****

**Young Researchers Club, Najafabad Branch, Islamic Azad University, Isfahan, Iran*
(E-mail: M.Taheri@sco.iaun.ac.ir)
***Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Isfahan, Iran*
(E-mail:{Naderi,Mehran_sharafi,Nadimi} @iaun.ac.ir)

Abstract: Robocup competition is an international event for research on fully autonomous robot control and related subject like: Artificial intelligence, Image processing, robot path planning, and obstacle avoidance. In this paper new practical software based methods for control, analysis, decision making and trajectory of an autonomous robot in Middle Size Soccer Robot league (MSL) are presented. In a robots soccer game, the environment is highly competitive and dynamic. In order to work in the dynamically changing environment, the software of a soccer robot system should have the features of flexibility, real-time control and adaptation. For this purpose, we utilize the sensor data fusion method in the control system parameters, self localization and world modelling. A vision-based self-localization and the conventional odometry systems are fused for robust self-localization. The methods have been tested in the many Robocup competition field middle size robots. This paper has tried to focus on description of areas such as omni directional mechanisms, omni-vision sensor for object detection, robot path planning, multimedia database management system for sophisticated offline analysis and other subjects related to mobile robot's software. The results are satisfactory which has already been successfully implemented in ADRO RoboCup team. This project is still in progress and some new interesting methods are described in the current report.

Keywords: Soccer Robot, Mobile robot, Machine vision, Omni directional movement, Multi agent, Robot path planning, Multimedia database management system

1. INTRODUCTION

Robotic soccer games had been popular with educational institutions around the world since the inauguration of the Robocup competition in 1997. This initiative provide a good platform for artificial intelligence, software control techniques and Multi Agent research, dealing with issues such as cooperation by distributed control, neural networks, genetic algorithms, decision tree, decision making, fuzzy logic [1]. In the context of RoboCup, the Middle Size League (MSL) is one of the most challenging. In this league, each team is composed of up to 5 robots with maximum size of 50x50cm base, 80cm height and a maximum weight of 40Kg, playing in a field of 18x12m. The rules of the game are similar to the official FIFA rules, with required changes to adapt for the playing robots . Each robot is autonomous and has its own sensorial means. They can communicate among them, and with an external computer acting as a coach, through a wireless network. This coach computer cannot have any sensor; it only knows what is reported by the playing robots. The agents should be able to evaluate the state of the world and make decisions suitable to fulfil the cooperative team objective. ADRO Middle Size project started in 2005, coordinated by the Electrical and Computer Department and involves many students working on several areas for building the mechanical structure of the robot, its hardware architecture and controllers and the software development in areas such as image analysis and processing, sensor and information fusion, reasoning and control. In 2007 we ranked 2nd place Middle Size Soccer Robot League in 2nd International China-Open Robocup Competitions, the China-Open is one of the, Asia's major RoboCup event. In 2008 we achieved the First Place in the 3rd International Iran-Open Competitions.

In this paper, at first we will describe the general hardware and software design of the ADRO Robocup team and after that focus on our scientific approaches in sensor fusion, learning and analysis, finally, concludes this paper [2][3].

2. HARDWARE ARCHITECTURE

Every fully autonomous robot of "ADRO" is equipped with an omni-directional vision system, a normal camera as front

vision, and an electromagnetic kicking device. The robot is controlled by a notebook PC is demonstrated in figure 1. The chassis of the robot is designed as a frame construction where there is the electronic circuit board, batteries, kicking device, motor controller and notebook PC. The omni-directional vision system and the normal camera are on the top of the framework. (fig.1)

Fig. 1 3-wheeled omni-directional mobile robot with an omni-directional vision system, a normal camera and a kicking device, and is controlled by a notebook PC.

2.1 Omni directional wheels

Omni directional robots usually use special wheels. These wheels are known as omni directional poly roller wheel. The omni-directional movement system consists of omni-directional wheels, DC motors, a drive shafting system and a controller. Although three such wheels are sufficient for the robot to move omni-directionally, a fourth wheel can provide redundancy in motion and control [1][3][4].

Our Robot structure includes three omni directional wheels for motion system and three small free wheels as feedback mechanism where shaft encoders are mounted on, as shown in figure 2. (fig.2)

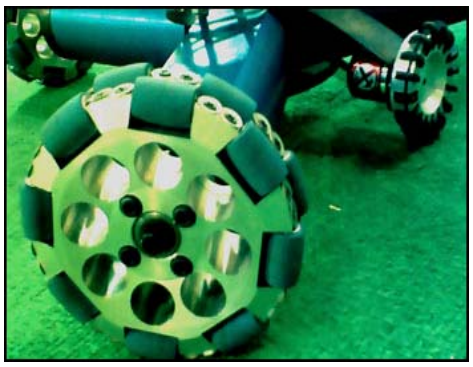

Fig.2 Three omni directional wheels act as actuators while three free wheels are for feedback.

2.2 Omni directional Vision system

Since the beginning of mobile robot manufacturing, the map building was one of the most addressed problems by researchers. Several researchers used Omni directional vision for robot navigation and map building [2]. Because of the wide field of view in Omni directional sensors, the robot does not need to look around using moving parts (cameras or mirrors) or turning the moving parts [5]. The Omni-directional vision system consists of a hyperbolic mirror, a firewire colour digital camera (Basler Digital Camera) and a regulation device. The mirror can make the resolution of the images of the objects near the robot on the field constant and make the distortion of the images of the objects far from the robot small in vertical direction (fig.3). Searching through different articles and catalogues from various mirror-making companies; we found that they used the following hyperbolic curve for their omni directional vision mirror [6].

$$\frac{x^2}{233.3} - \frac{y^2}{1135.7} = -1 \qquad (1)$$

However, this equation is suitable for the mirror with large size and wide view. For our soccer player robot, we need an image with a diameter of 4m on the field, so to achieve a compact mirror with wide view, the above curve scaled down by a factor of 2.5 to yields:

$$\frac{x^2}{233.3} - \frac{y^2}{1135.7} = -6.43 \qquad (2)$$

Fig 3. hyperbolic mirror and the typical panoramic image captured.

3.ROBOT KINEMATICS

Using omni directional wheels, the schematic view of robot kinematics can be shown as follows (fig.4) [6], where is the robot center of mass, P_O is defined as the vector connecting O to the origin and D is the drive direction vector of each wheel.

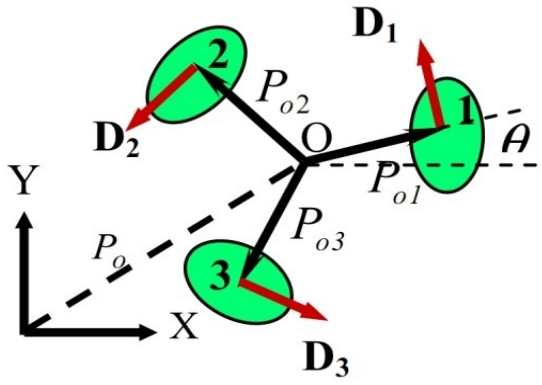

Fig. 4 Robot kinematics diagram

Using unitary rotation matrix, $R(\theta)$ defined as:

$$R(\theta)=\begin{bmatrix}\cos(\theta) & -\sin(\theta)\\ \sin(\theta) & \cos(\theta)\end{bmatrix} \quad (3)$$

The positions vectors $P_{O1},...,P_{O3}$ with respect to the local coordinates centered at the robot center of mass are given as:

$$P_{O1}=L\begin{bmatrix}1\\0\end{bmatrix} \quad (4)$$

$$P_{O2}=R(\frac{2\pi}{3})*P_{O1}=\frac{L}{2}\begin{bmatrix}-1\\\sqrt{3}\end{bmatrix} \quad (5)$$

$$P_{O3}=R(\frac{4\pi}{3})*P_{O1}=-\frac{L}{2}\begin{bmatrix}1\\\sqrt{3}\end{bmatrix} \quad (6)$$

The drive directions can be obtained by:

$$D_i=\frac{1}{L}R(\theta)*P_{Oi}, \quad (7)$$

$$D_1=\begin{bmatrix}0\\1\end{bmatrix},\quad D_2=\begin{bmatrix}1\\0\end{bmatrix},\quad D_3=\frac{1}{2}\begin{bmatrix}\sqrt{3}\\-1\end{bmatrix}, \quad (8)$$

Where, L is the distance of wheels from the robot center of mass (O).

Using the above notations, the wheel position and velocity vectors can be expressed with the use of rotation matrix $R(\theta)$ as:

$$\begin{aligned}R_i&=P_O+R(\theta)*P_{Oi},\\ V_i&=\dot{P}_O+\dot{R}(\theta)*P_{Oi},\end{aligned} \quad (9)$$

The vector $P_O=\begin{bmatrix}X & Y^T\end{bmatrix}$ is the position of the center of mass with respect to Cartesian coordinates.

The angular velocity of each wheel can be expressed as:

$$\phi_i=\frac{1}{r}V_i^T*R(\theta)*D_i \quad (10)$$

Where, r is the system wheel radius of odometry.

Substituting for V_i from equation (9) yields:

$$\phi_i=\frac{1}{r}\left[P_O^T*R(\theta)*D_i+P_{Oi}*\dot{R}(\theta)^T*R(\theta)*D_i\right] \quad (11)$$

Note that the second term in the right hand side is the tangential velocity of the wheel.

On the other hand, this tangential velocity is equal to:

$$L\dot{\theta}={P_{Oi}}^T*\dot{R}(\theta)*R(\theta)*D_i \quad (12)$$

From the kinematics model of the robot, it is clear that the wheel velocity is a function of linear and angular velocities of robot center of mass, i.e.:

$$\begin{Bmatrix}\dot{\phi}_1\\\dot{\phi}_2\\\dot{\phi}_3\end{Bmatrix}=\frac{1}{r}\begin{bmatrix}-\sin(\theta) & \cos(\theta) & L\\ -\sin(\frac{\pi}{3}-\theta) & -\cos(\frac{\pi}{3}-\theta) & L\\ \sin(\frac{\pi}{3}+\theta) & -\cos(\frac{\pi}{3}+\theta) & L\end{bmatrix}\begin{Bmatrix}\dot{X}\\\dot{Y}\\\dot{\theta}\end{Bmatrix} \quad (13)$$

or in short:

$$\dot{\varphi}=\frac{1}{r}W*\dot{S} \quad (14)$$

where L is the distance of wheels from the robot center of gravity (O) and r is the main wheel radius [6],[7].

3. ROBOT DYNAMICS

Linear and angular momentum balance for the robot may be written as:

$$\begin{aligned}\sum_{i=1}^{3}f_iR(\theta)*D_i&=m\ddot{P}_O,\\ L\sum_{i=1}^{3}f_i&=J\ddot{\theta}\end{aligned} \quad (15)$$

where $\ddot{P}_O$ is the acceleration vector, f_i if is the magnitude of the force produced by the *i*th motor, *m* is the mass of the robot and *J* is its moment of inertia about its center of gravity. Assuming no-slip condition, the force generated by a DC motor is described by:

$$f=\alpha U+\beta V \quad (16)$$

where, $V=\{V_i(t),i=1,2,3\}$ is the velocity of each wheel. The constants α and β are motor characteristic coefficients and can be determined either from experiments or from motor catalogue.[8]

Note that $U=\{U_i(t),i=1,2,3\}$ is the voltage applied by supplier to the DC motors. Substituting equation (13) into equation (12) yields:

$$\begin{aligned}\sum_{i=1}^{3}(\alpha U_i-\beta V_i)R(\theta)D_i&=m\ddot{P}_O,\\ L\sum_{i=1}^{3}(\alpha U_i-\beta V_i)&=J\ddot{\theta}\end{aligned} \quad (17)$$

This system of differential equations may be written in the matrix form as:

$$\begin{bmatrix}m\ddot{X}\\m\ddot{Y}\\m\ddot{\theta}\end{bmatrix}=\alpha P(\theta)U-\frac{3\beta}{2}\begin{bmatrix}\dot{X}\\\dot{Y}\\2L^2\dot{\theta}\end{bmatrix} \quad (18)$$

$$P(\theta)=\begin{bmatrix}-\sin(\theta) & \cos(\theta) & L\\ -\sin(\frac{\pi}{3}-\theta) & -\cos(\frac{\pi}{3}-\theta) & L\\ \sin(\frac{\pi}{3}+\theta) & -\cos(\frac{\pi}{3}+\theta) & L\end{bmatrix}^T \quad (19)$$

and

$$U=\begin{bmatrix}U_1(t) & U_2(t) & U_3(t)^T\end{bmatrix} \quad (20)$$

4.ROBOT SOFTWARE

We have developed a software system to fully utilize the hardware abilities. In this section introduces software parts contain: image processing algorithm, position controller architecture, world model construction, artificial intelligence, trajectory, and network and team strategy from a viewpoint of software system. (fig.5)

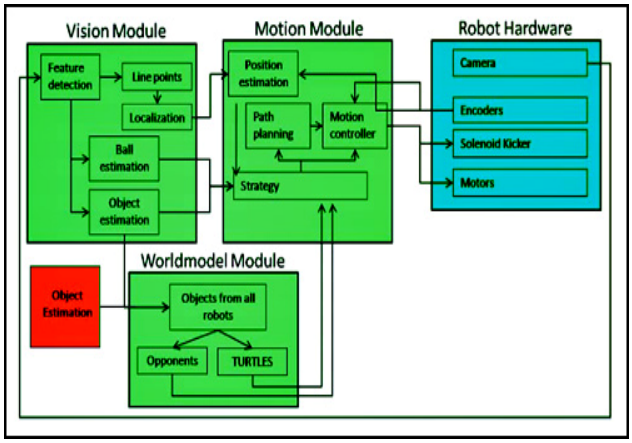

Fig 5. Block diagram of the Robot Software

During the dribble, the robot adjusts its direction toward the opponent goal and dribbles with the fastest possible speed. The support robot takes a position behind and near to the robot with the ball. The support robot fetches the ball only when the ball is near to the support robot. The defence robot is located between the ball and own goal. The defence robot doesn't actively approach when the ball is far. In our team strategy three states are allotted to the team robots: attack, defence, and intercept. The robots autonomously choose to activate each of the roles [9]. (fig.6)

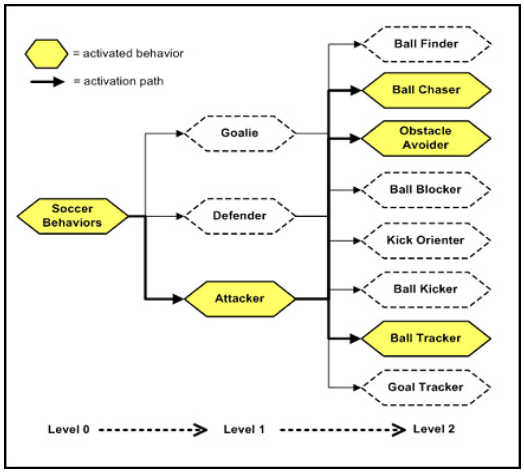

Fig 6. Behaviour hierarchy suggested for robot soccer

4.1 Image processing algorithm

Utilizing a digital camera, each time the computer on each robot performs the processing of the current frame and calculates the position, direction and velocity of the robot. It also determines the position and velocity of the opponent robots as well as the position and velocity of the ball (fig.7). The algorithm used to find objects is optimized to process the maximum number of frames. First it searches the pixels by swiping them with certain steps, when it finds a desired one and detects that object, saves its coordinates so the next time it can start back with the same point about. We are trying to evaluate new methods to find some kinds of objects based on pattern recognition to reduce the effect of changing the colours on algorithm. The image processor receives its data through fire wire port connected to a Basler digital video camera with the speed of 20 to 30 frames per second [1][5].

Fig 7. Vision Systems, Object detection (Goals, Flag Spots, Ball)

4.2 World model construction

Although each agent tries to extract the real world map as accurate as possible, but "noisy data" and "non-global optimized" algorithms reduce the reliability of processed data. The world model module receives different data sets from every agent. Each data set contains different environmental information like self, ball and opponents' positions. Each data carries a 'confidence' factor; a larger confidence factor means a more reliable piece of information. The most recent data sets are then chosen for data fusion, in which the following rules and facts are applied:

- Closer object are of more accuracy.
- Objects further than a specific distance could be said to be totally inaccurate. (This distance is experimentally known)

This new world model contains information about the objects which may not have been seen by each agent correctly and also enhances approximations of all environmental information. The constructed world model is then sent back to all agents so they will have a better view of the world around them.

The interaction between the modules on different machines is provided by a communication protocol which bundles commands and parameters generating command packets and interprets the incoming packets for other modules. In the

following, each layer, its interface and parameters will be discussed in details [10][11]. (fig.8)

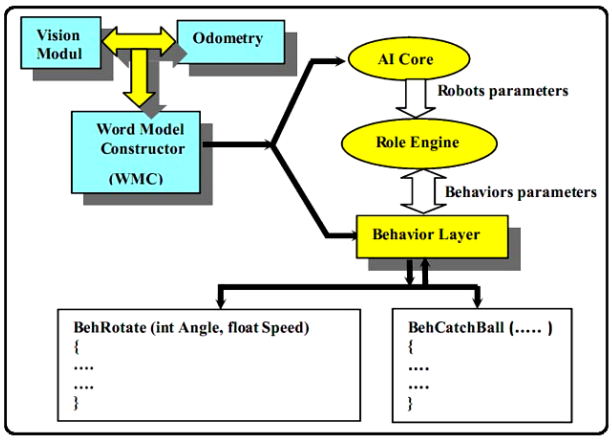

Fig 8. World model construction and artificial intelligent structure

4.3 Artificial intelligence

In this section the AI part of the software is briefly introduced. There are three distinct layers: AI Core, Role Engine and Behaviour Engine. AI Core receives the computed field data from world modelling unit and determines the play state according to the ball, opponents and our robots positions. Considering the current game strategy, determination of the play state is done by fuzzy decision-making to avoid undesirable and sudden changes of roles or behaviours. Then AI Core sends a set of roles to Role Engine to be assigned to the robots. Because there are instances in which the image-processing unit cannot see the ball, a memory is implemented in the AI Core for the position of ball that specifies which robot owns the ball. Since there is a relationship between new roles and old roles, roles are changed in a manner that robots never experiment sudden changes in roles (for example the role never changes from defence to attack in next cycle). Role Engine receives a set of roles from AI Core and provides the Behaviour Engine with a set of behaviours for robots. Twin or triple roles are implemented so that the robots really cooperate with each other to do their roles. Behaviours are the building blocks of the robot's performance which includes simple actions like rotating, or getting the ball and etc. The Behaviour Layer is the lowest layer in our architecture. This layer receives a sequence of behaviours along with some parameters from the upper layer (Role Engine) and executes the essential subroutines in order to accomplish certain behaviour. These subroutines use world model information and trajectory data in order to perform necessary movements [12].

4.4 Robot Self Localization

Our self localization method is based on detection of white lines in field. Because according to MSL rule no flag and no colour goals exist in field since Robocup 2008, now the white line points are the only visual information that could be used as landmarks for robot's self localization. So our vision system tracks all white lines that exist only in the region field colour (green) and robots use a digital compass (MTi-sensor) for the robot heading reference. After this section we try to convert the acquired white line point into the real world distance map. (fig.9)

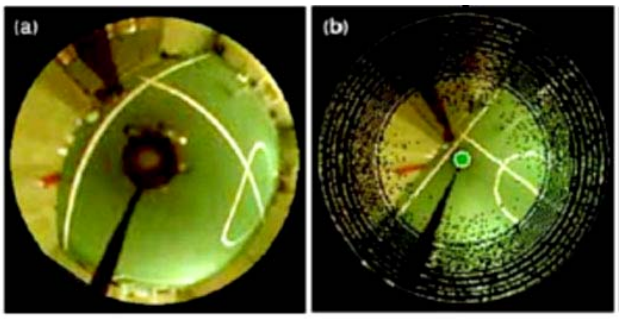

Fig 9: The field lines detection and self localization.

We employ a Monte Carlo Localization (MCL) method. For our algorithm, the field model is a Cartesian coordinate system with the origin at the centre of the field. The robot's state is represented by a vector $X=[x,y,\theta]^T$ which consists of a position (X,Y) and θ an orientation. We provided the algorithm, which detects only orientation, made the posture θ ingredient known in MCL, and planned the dimension reduction of the state vector. The orientation detection is explained further below. For localizing, we have to construct the posterior density $p(x_t \mid y_1 ... y_t)$ from the state of a robot x_t and the sensor data y_t at the current time t. In the particle filter methods, a probability density is represented by a set of N random samples (Particle). The method proceeds in two phases.

In the first phase we predict a current state of the robot. That is specified as a conditional density $p(x_t \mid x_{t-1}..u_{t-1})$ from the previous state x_{t-1} and a control input u_{t-1}. The predictive density is obtained by the following integral. For our algorithm, we set the control input u_{t-1} as odometry data and add it to each particle.

$$p(x_t \mid y_1 ... y_t) = \int p(x_t \mid x_{t-1}, u_{t-1}) p(x_{t-1} \mid y_1 ... y_{t-1}) dx_{t-1} \qquad (21)$$

In the second phase we update the density according to the sensor data y_t . The likelihood of y_t at state x_t is represented as $p(y_t \mid x_t)$. The posterior density is obtained using Bayes theorem.

$$p(x_t \mid y_1 ... y_t) = \frac{p(y_t \mid x_t) p(x_t \mid y_1 ... y_{t-1})}{p(y_t \mid y_1 ... y_{t-1})} \qquad (22)$$

Sensor data y_t is distance to the field line. The state is compared to y_t and the likelihood is updated of each particle. After that, weighted particles are normalized and re-sampled. Re-sampling is done according to the weight of each particle: new particles are generated around the particles that have high likelihood. In the Robocup soccer field most constituents are straight lines or perpendicular

segments of lines. The robot's orientation is detected by searching for inclination of the straight line ingredients in the circumference seen from the robot. We solved the false detection by using compass sensor for this problem. Our self localization algorithm is a one of the very fast and effective algorithm to track robot's localization, and it only takes several milliseconds to finish the localization computation for one frame image. Adro has developed and implemented three separate Omni directional wheels coupled with shaft encoders placed 60 ° apart of the main driving wheels. Free shaft rotation and the flexible connection to the structure ensures minimum slippage and firm contact of these wheels to the ground, all these result in a great improvement in output precision. In order to avoid the remaining cumulative error, odometry system parameters can be initialized every time the vision could calculate the position reliably [13][14].

4.5 Trajectory

Since the motion trajectory of each robot is divided into several median points that the robot should reach them one by one in a sequence the output obtained after the execution of AI will be a set of position and velocity vectors. So the task of the trajectory will be to guide the robots through the opponents to reach the destination. The routine used for this purpose is the potential field method (also an alternative new method is in progress which models the robot motion through opponents same as the flowing of a bulk of water through obstacles) [14]. In this method different electrical charges are assigned to our robots, opponents and the ball. Then by calculating the potential field of this system of charges a path will be suggested for the robot. At a higher level, predictions can be used to anticipate the position of the opponents and make better decisions in order to reach the desired vector. In our path planning algorithm, an artificial potential field is set up in the space; that is, each point in the space is assigned a scalar value. The value at the goal point is set to be 0 and the value of the potential at all other points is positive. The potential at each point has two contributions: a goal force that causes the potential to

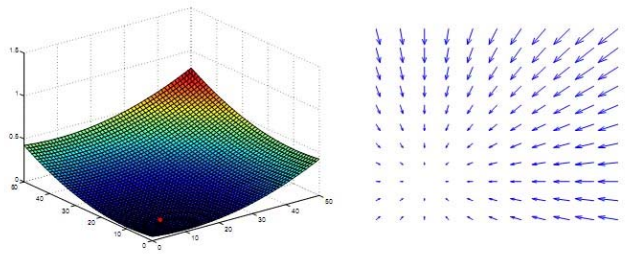

Fig 10. Goal force (Attractive potential to the goal)

increase with path distance from the goal, and an obstacle force that increases in inverse proportion to the distance to the nearest obstacle boundary. In other words, the potential is lowest at the goal, large at points far from the goal, and large at points next to obstacles. If the potential is suitably defined, then if a robot starts at any point in the space and always moves in the direction of the steepest negative potential slope, then the robot will move towards the goal while avoiding obstacles. The numerical potential field path planner is guaranteed to produce a path even if the start or goal is placed in an obstacle.

$$U(q) = U_{Goal}(q) + U_{Obstacle}(q) \quad (23)$$

If there is no possible way to get from the start to the goal without passing through an obstacle then the path planner will generate a path through the obstacle, although if there is any alternative then the path will do that instead. For this reason it is important to make sure that there is some possible path, although there are ways around this restriction such as returning an error if the potential at the start point is too high. The path is found by moving to the neigh boring square with the lowest potential, starting at any point in the space and stopping when the goal is reached [8][15][16]. (fig.10)

4.6 Multimedia database

In Dynamic environment, some of images, video and audio saves in multimedia database to use in offline analysis for better detecting objects. Various architectures are being examined to design and develop a Multimedia Database Management System (MMDBMS). In our approach, the architecture, illustrated in Figure 11, is the tight coupling, that the DBMS manages both the multimedia databases as well as the metadata (fig.11). This includes query processing, transaction management, metadata management, storage management, and security and integrity management.

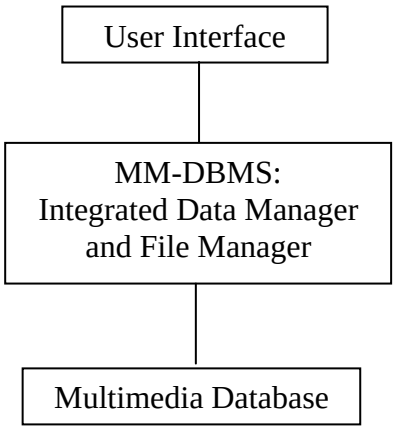

Fig 11. Tight coupling approach

Image processing has dealt with areas such as detecting abnormal patterns which deviate from the norm, retrieving images by content, and pattern matching (fig.12). The main question here is what in image mining? How does it differ from image processing? One can say that while image processing is focusing on detecting abnormal patterns as well as retrieving images, image mining is all about finding unusual patterns. Therefore, one can say that image mining deals with making associations between different images from large image databases. Note that detecting unusual patterns is not the only outcome of image mining. There has

been work to identify recurring themes in images, both at the level of raw images and with higher-level concepts. Mining video data is even more complicated than mining image data. One can regard video to be a collection of moving images, much like animation. The important areas include developing query and retrieval techniques for video databases, including video indexing, query languages, and optimization strategies [17][18]. (fig.13)

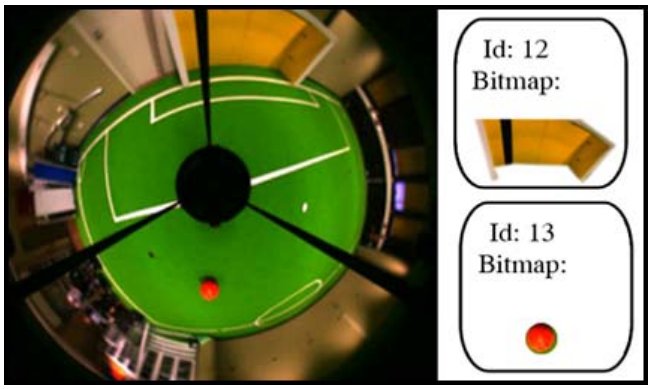

Fig 12. Iconic data with semantic and database objects

There are two key advances needed before multimedia data mining will become a reality: 1- mining techniques that model order as a key part of the data. Too much is lost when the sequence of multimedia is ignored. The two approaches to mining ordered data, time series and event sequences, are not adequate for multimedia.

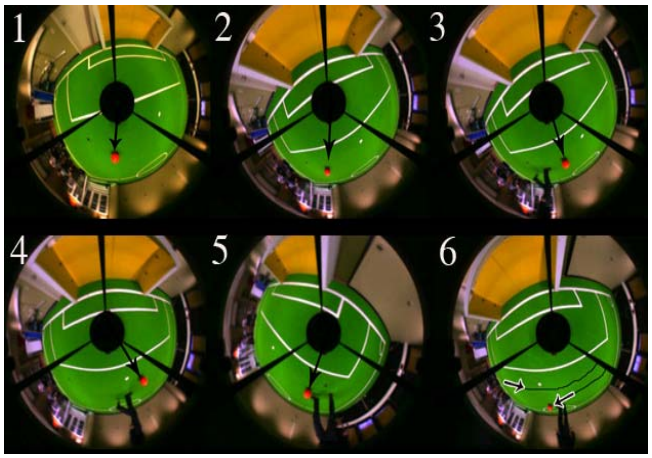

Fig 13. Event-Based Video Data

One idea is to capture order in the results, e.g., discovered patterns could include "pattern1 before pattern2". 2- The ability to compare objects that are represented differently. Pictures taken from different angles, or a photograph and a line drawing, both capture similar information. However, the different representations will lead data mining algorithms to overestimates the differences between the objects. If an algorithm recognizes many similarities in the data about two objects, it is likely that the objects are similar. However, dissimilar data does not imply that objects are different-differences between the ways data is captured may cause similar objects to be represented by very different sets of data. Mining techniques must handle this disparity [19][20]. (fig.14)

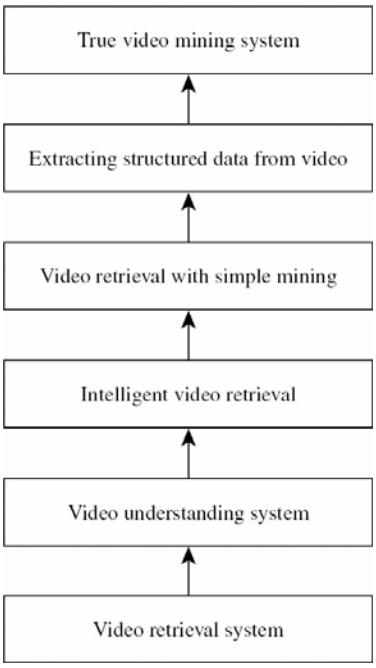

Fig 14. Video mining system

5. CONCLUSION

The performance of our robot team in Iran-Open Robocup competitions (1st place) (fig.15) showed that the combination of methods and techniques described in this paper are led to a successful soccer player team. In our robot, omni directional navigation system, omni-vision system and a novel control and analysis based on MMDBMS have been combined to create a comprehensive omni directional robot. The idea of separating odometry sensors from the driving wheels was successfully implemented. Three separate omni directional wheels coupled with shaft encoders placed apart of the main driving wheels. The result was reducing errors such as slippage in the time of acceleration. Combination of odometry and vision led to a more accurate and reliable self-localization algorithm.

Fig 15. ADRO middle size soccer robot team in Robocup competition

REFERENCES

[1] Neves, A. J. R., A. J. Pinho, et al.. "An efficient omnidirectional vision system for soccer robots: From calibration to object detection." Mechatronics 21(2): 399-410, 2011.

[2] M.Taheri, S.H. Mohades Kasaei, S.M.Mohades Kasaei, M.Delshad and N.Ahmadi, “A Practical Approach to Dynamic Role Engine and Formation Control for a middle size soccer robot team” 12th international Conference on climbing and walking robot and the support technologies for mobile machines 9-11 September 2009, Istanbul, Turkey.

[3] M.Taheri, S.H.Mohades Kasaei, S.M.Mohades Kasaei, "Development of a Soccer Robot and Autonomous Navigation System", 1st RoboCup IranOpen International Symposium, Qazvin, Iran, April 3-6, 2009.

[4] Takemura, Y., Y. Ogawa.. "A System Design Concept Based on Omni-Directional Mobility, Safety and Modularity for an Autonomous Mobile Soccer Robot." Journal of Bionic Engineering 5(Supplement 1): 121-129, 2008.

[5] Lu, H., S. Yang, et al. "A robust omnidirectional vision sensor for soccer robots." Mechatronics 21(2): 373-389, 2011.

[6] Alexander Gloye, Ra´ul Rojas. “Robot Heal Thyself: Precise and Fault-Tolerant Control of Imprecise or Malfunctioning Robots”. RoboCup 2005 International Symposium, Osaka, japan, 2005.

[7] Li, K. and R. D'Andrea. "Motion design and learning of autonomous robots based on primitives and heuristic cost-to-go." Robotics and Autonomous Systems 56(8): 658-669, 2008.

[8] Martins, F. N., W. C. Celeste, et al. "An adaptive dynamic controller for autonomous mobile robot trajectory tracking." Control Engineering Practice 16(11): 1354-1363, 2008.

[9] Lau, N., L. S. Lopes, et al. "Robot team coordination using dynamic role and positioning assignment and role based setplays." Mechatronics 21(2): 445-454, 2011.

[10] López, J., D. Pérez, et al. "A framework for building mobile single and multi-robot applications." Robotics and Autonomous Systems 59(3-4): 151-162, 2011.

[11] Silva, J., N. Lau, et al. "World modeling on an MSL robotic soccer team." Mechatronics 21(2): 411-422, 2011.

[12] Posadas, J. L., J. L. Poza. "Agent-based distributed architecture for mobile robot control." Engineering Applications of Artificial Intelligence 21(6): 805-823, 2011.

[13] Tamimi, H., H. Andreasson. "Localization of mobile robots with omnidirectional vision using Particle Filter and iterative SIFT." Robotics and Autonomous Systems 54(9): 758-765,2006.

[14] Tesng, C. S., Chen, B.S., and Uang, H.J., “Fuzzy Traking Control for Nonlinear System via T-S Fuzzy Model,” IEEE Trans. Fuzzy system, Vol. 9 , 2001, pp. 381.

[15] Liu, Y., J. J. Zhu.. "Omni-directional mobile robot controller based on trajectory linearization." Robotics and Autonomous Systems 56(5): 461-479, 2008.

[16] Yin, L., Y. Yin, et al. "A new potential field method for mobile robot path planning in the dynamic environments." Asian Journal of Control 11(2): 214-225, 2009.

[17] Döller, M. and H. Kosch. "The MPEG-7 Multimedia Database System (MPEG-7 MMDB)." Journal of Systems and Software 81(9): 1559-1580, 2008.

[18] Shyu, M.-L. and S.-C. Chen. "Introduction to the special issue on multimedia databases." Information Systems 31(7): 636-637, 2006.

[19] Zhang, T. and H. Ueno. "Knowledge model-based heterogeneous multi-robot system implemented by a software platform." Knowledge-Based Systems 20(3): 310-319, 2007.

[20] Sevilmis, T., M. Bastan, et al. (2008). "Automatic detection of salient objects and spatial relations in videos for a video database system." Image and Vision Computing 26(10): 1384-1396, 2008.