World Academy of Science, Engineering and Technology
International Journal of Computer and Systems Engineering
Vol:1, No:6, 2007

Design of an Artificial Intelligence Based Automatic Task Planner or a Robotic System

T. C. Manjunath, C. Ardil

***Abstract*—**This paper deals with the design and the implementation of an automatic task planner for a robot, irrespective of whether it is a stationary robot or a mobile robot. The aim of the task planner nothing but, they are planning systems which are used to plan a particular task and do the robotic manipulation. This planning system is embedded into the system software in the computer, which is interfaced to the computer. When the instructions are given using the computer, this is transformed into real time application using the robot. All the AI based algorithms are written and saved in the control software, which acts as the intelligent task planning system.

***Keywords*—**AI, Robot, Task Planner, RT, Algorithm, Specs, Controller.

I. Introduction

ONE of the most important problems in robotics is the task-planning problem. A task is a job or an application or an operation that has to be done by the robot, whether it is a stationary robot or a mobile robot. The word planning means deciding on a course of action before acting. Before a robot does a particular task, how the task has to be done or performed in its workspace has to be planned. This is what is called as Robot Task Planning [2]**.** The tasks will be usually specified by the user and is therefore considered as a problem that has to be given to the task planner [1].

A plan is a representation of a course of action for achieving the goal. How the problem has to be solved has to be planned properly. Robot task planning is also called as problem solving techniques and is one of the important topics of Artificial Intelligence. For example, when a problem is given to a human being to be solved; first, he or she thinks about how to solve the problem, and then devises a strategy or a plan how to tackle the problem. Then only he or she starts solving the problem. Hence, robot task planning is also called as robot problem solving techniques. Many of the sub-topics in this chapter are currently under active research in the fields of Artificial Intelligence, Image Processing and Robotics.

Dr. T.C. Manjunath has a Ph.D. from the renowned Systems and Control Engineering Department of the Indian Institute of Technology Bombay, Post-graduate from LD College, Gujarat Univeristy & Graduate from RVCE, Bangalore University.
E-mail: tcmanjunath@rediffmail.com, tcmanjunath@gmail.com.
Cemal Ardil is with the National Academy of Aviation, AZ 1056 Baku, Azerbaijan.

Lot of research is going on in the robot problem solving techniques. A typical robot problem solving consists of doing a household work; say, opening a door and passing through various doors to a room to get a object. Here, it should take into consideration, the various types of obstacles that come in its way; also the front image of the scene has to be considered the most. Hence, robot vision plays a important role. In a typical formulation of a robot problem, we have a robot that is equipped with an array of various types of sensors and a set of primitive actions that it can perform in some easy way to understand the world.

Robot actions change one state or configuration of one world into another. For example, there are several labeled blocks lying on a table and are scattered. A robot arm along with a camera system is also there. The task is to pick up these blocks and place them in order [4]. In a majority of the other problems, a mobile robot with a vision system can be used to perform various tasks in a robot environment containing other objects such as to move objects from one place to another ; i.e., doing assembly operations avoiding all the collisions with the obstacles [3].

The paper is organized in the following sequence. A brief introduction about the task planner and the task planning system was given in the previous paragraphs. The section gives information about the task planner design. Section 3 gives the information about the inputs-outputs of the task planner. Section 4 gives information about the brief explanation of the design of the task planner. This is followed by the conclusions in section 5 and the references.

II. Physical System Design & A Brief Report

The mechanical design of the developed system is divided into 3 parts, viz., the base assembly, the arm assembly and the gripper assembly. The designed robot has R-R-P (Rotary-Rotary-Prismatic) type of axes [2]. A four axis / four DOF designed SCARA robot arm as shown in Fig. 1. A SCARA robot is a 4 DOF stationary robot arm having base, elbow, vertical extension and tool roll and consisting of both rotary and prismatic joints. There is no yaw and pitch, only roll. There are 4 joints, 4 axis (three major axes - base, elbow, vertical extension and one minor axis - tool roll) [4]. The 4 DOF's are given by Base, Elbow, Vertical Extension and Roll, i.e., there are three rotary joints and one prismatic joint. Since n = 4; 16

KP's are to be obtained and 5 RHOCF's are to be attached to the various joints [3]. The task planner is designed for the stationary robotic system shown in the Fig. 1.

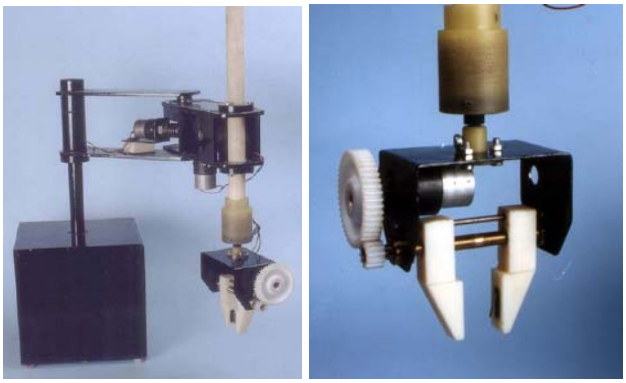

Fig. 1 The designed SCARA robot

III. Design of the Task Planner

In a highly sophisticated robot environment, such as in automatic production plants, assembly lines, higher level planning of robot motion is required in order to extract the full benefits of the automation [5]. The planning at this level is called as task planning. Task planning deals with the goals of the manipulation task rather than how to achieve these goals [6]. The goal is the destination or the end of the task, i.e., the desired configuration of the part in the place position. The robot task planning requires task planners to achieve these goals. Planning should be certainly regarded as an intelligent function of any robot. The task planners are defined as planning systems, which are used to plan various types of robot motions and to do a particular task [7]. The task planner makes use of simulation software to do the particular task. It is a black box as shown in the Fig. 2 below.

The input to the task planner is known as task specifications and the output of the task planner is various types of robot motions that are needed to do the particular task. The task planner is similar to the automatic program generators in artificial intelligence [8]. The input to the generator is the task to be specified. The output of the generator is various types of solutions. The task specifications are to be supplied by the user and from the robot vision system [9]. The task planner requires a large amount of external and internal information (data) in order to do a particular task. These data's or information's includes the following [10]:

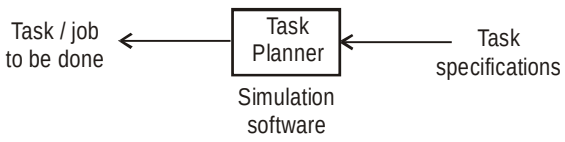

Fig. 2 Black box of a task planner

IV. Inputs / Outputs of the Task Planner

The inputs and outputs of the task planners are shown below as follows [10].

- Details of the internal / external environment of the robot,
- Position of the objects and the obstacles in the 3D Euclidean space,
- Scene (the foreground and the background scene) which is obtained by the robot vision system,
- On-line feedback data from the sensors,
- Position of the arm (joints) in the intermediate stage,
- Description of the objects being manipulated,
- Task environment, i.e., the environment in which the task is to be performed (say, a nuclear reactor environment - in which the robot has to handle nuclear waste material, steel industry - to lift molten iron steel, space explorations - where the robot has move on the rocky terrain of the MARS and pick up small rock samples and take pictures and send them to earth) [11].
- The initial state of the environment (source - the pick position),
- The final (goal - destination - the place position) state of the environment.

The relationships between a task planner and the other parts of the robotic system are shown in the block-diagram in the Fig. 4. A robot task planner attempts to find a path from the initial robot world (pick position) to a final robot world (place position). The path consists of a sequence of operations that are considered primitive to the system [10].

A task planner devises a plan for a complex task as a sequence of simple actions called "atomic actions", which the robot knows how to execute without further planning [2]. For example, if a human gives an order to a robot such as "bring me a cup of water," then the robot's task planner should decompose the task as follows [12]

- find the location of the cup of water,
- plan a path and navigate to the cup
- grasp the cup
- plan a path and navigate to the user,
- hand over the cup to the user.

V. Explanation of the Design of TP

Although a general-purpose inference engine could be used to solve the task planning problem, it requires a huge body of knowledge before it can plan action sequences for a variety of household tasks. We instead take a procedural knowledge approach to planning a set of frequently used commands such as "bring an object to a destination," and store the action sequences explicitly in a knowledge base [13]. We do this for as many tasks as we can manage. For the tasks not in the knowledge base, then, an instance-based learning algorithm to find the most similar tasks and generate a plan by modifying the plans for the similar tasks can be used [14].

First, the user just places the object at the pick position and tells to the robot vision system, look, this is the starting point, you yourself have to find the position and orientation of the object at this pick point and pick up the object [15]. The vision system, clicks the source scene using a camera and using image analysis, finds the position and orientation of the object at the pick point and other information such as the foreground and the background in the source scene and stores all these information in the memory of the computer in the form of a database. Now, the robot knows the starting position of the object [16].

Next, the user keeps the object at the place point and tells to the robot vision system, look, this is the destination position, you yourself have to find the position and orientation of the object at the place position and place the object is the desired position and orientation [17]. The vision system, then clicks the scene using the camera and using image analysis finds the position and orientation of the object at the place point and other information such as the foreground and the background in the goal scene and stores the information collected in the memory of the computer in the form of a database. Now, the robot knows the final position of the object [18].

Once, we specify the pick and the place point, the robot knows how the object has to be picked up and how it has to be placed, since it is already knowing both the pick position and the place positions. This information, which is stored in the memory in the form of a database, is given as input to the task planner as the task specifications [19]. The raw images (analog images) which are captured by the overhead cameras are first processed by the image analyzers and the raw image data is converted into a digital data (data base) which is of a form that can be easily processed by the task planner software. The task planner gives 'n' number of solutions (i.e., 'n' number of programs, since inverse kinematics is not unique), out of which it selects one optimized solution, i.e., the shortest path solution and gives it as input to the trajectory planner and the controller [20].

These output programs would be various types of robot motions in the tool configuration space R^6. Once a movement is planned in the form of a discrete tool-configuration trajectory $[w^k]$, this information is sent to the trajectory planner. The trajectory of the tip of the robot arm will be planned. The trajectory planner makes use of various types of interpolation techniques [such as cubic polynomial technique, linear interpolation, blended trajectory technique, straight line motion techniques] and the inverse kinematics equations to convert the discrete tool-configuration trajectory $[w^k]$ into an equivalent continuous-time joint-space reference trajectory r(t).

The joint space trajectory is given as input to the robot controller, which contains a torque regulator and generates the required torque profile $\tau(t)$ for the joints of the robot arm. The controller gives a control signal to move the robot arm from the source to the goal. During this stage, the arm kinematics, arm dynamics comes into picture. Force, torque, moment, proximity and tactile sensors can be mounted on the robot arm and these sensors can be used to provide the feedback data F^{tool} from the sensors to the robot controller [21]. The robot controller makes use of these data to avoid the collision of the robot with the obstacles, implement the guarded motion and compliant motion and thus complete the manipulation task successfully avoiding all the obstacles in its path of motion. Also, the task environment should be specified.

By virtue of their versatility, robots can be difficult to program, especially for tasks requiring complex motions involving sensory feedback. In order to simplify programming, task-level languages exist that specify actions in terms of their effects on objects. Example: pin A programmer should be able to specify that the robot should put a pin in a hole, without telling it what sequence of operators to use, or having to think about its sensory or motor operators. Task planning is divided into three phases: modeling, task specification, and manipulator program synthesis. There are three approaches to specifying the model state: Using a CAD system to draw the positions of the objects in the desired configuration [22].

Using the robot itself to specify its configurations and to locate the object features. Using symbolic spatial relationships between object features (such as (face1 against face2). This is the most common method, but must be converted into numerical form to be used. One problem is that these configurations may over-constrain the state. Symmetry is an example; it does not matter what the orientation of a peg in a hole is. The final state may also not completely specify the operation; for example, it may not say how hard to tighten a bolt. The three basic kinds of motions are free motion, guarded motion, and compliant motion. An important part of robot program synthesis should be the inclusion of sensor tests for error detection [23].

After a user's input command, the robot's task planner analyzes various attributes of the given task and searches similar cases by exploring the knowledge base. If there exists a matching task with enough similarity to the given task, the preplanned action sequence of the matching task is appropriately modified to generate a plan to carry out the given task. If the new plan has enough novelty with respect to all the existing plans, it is added to the knowledge base, which constitutes a learning process of the task planner. When there is no similar task, the robot asks the user to demonstrate how to perform the task (using text input at present) and stores the action sequence taught by the user. This constitutes the second learning process of the task planner as shown in the Fig. 3.

A basic problem in robotics is to resolve specified tasks and commands and plan the resulting motions and sub-tasks. The planning system needs to transform a task-oriented problem into a plan, which describes how the given problem can be

solved by the robot. For this transformation a detailed knowledge base and world model have to be available. These models give the robot a description of its environment, and thus enable it to construct the necessary operations needed to fulfill the task. The plan generated in this way contains a sequence of action elements (e.g. movement, picking up items, manipulating items) with assigned resources (e.g. the robot or its gripper) [9]. Then, these sequences of actions are transformed into the reality stage using the real time implementation software and the necessary interfaces.

The motion control manager determines the start and destination of a path and plans the course and the necessary actions. The resulting motions of a robot are called trajectories or paths and consist of a sequence of desired positions, velocities and accelerations at a given point. The sequence of plan elements is called a task execution sequence. During this planning stage all constraints and restrictions, like closed or impassable areas are considered as well as target times, resources, supplies or the processing of parallel or sequential tasks as shown in the Figs. 4 and 5 respectively [9].

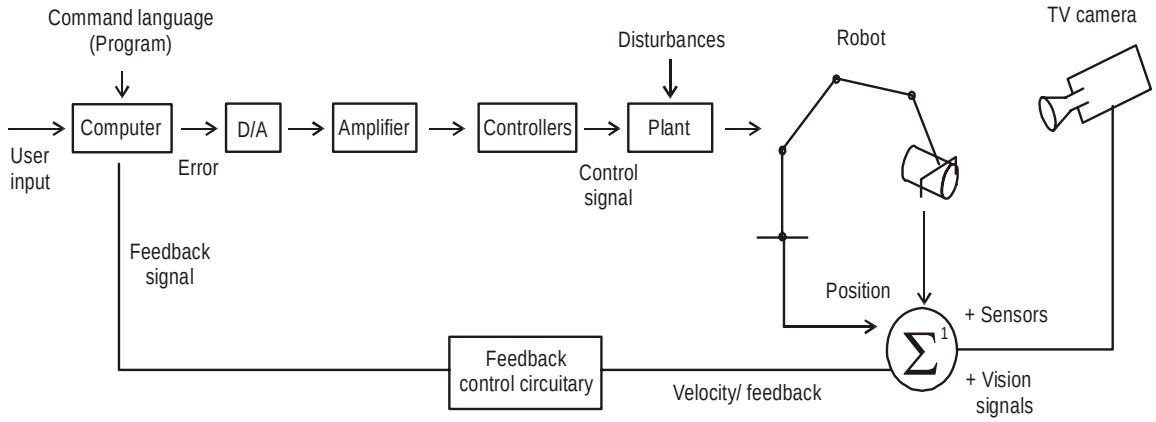

Fig. 3 Block diagram of the robot control system interfaced to the task planner

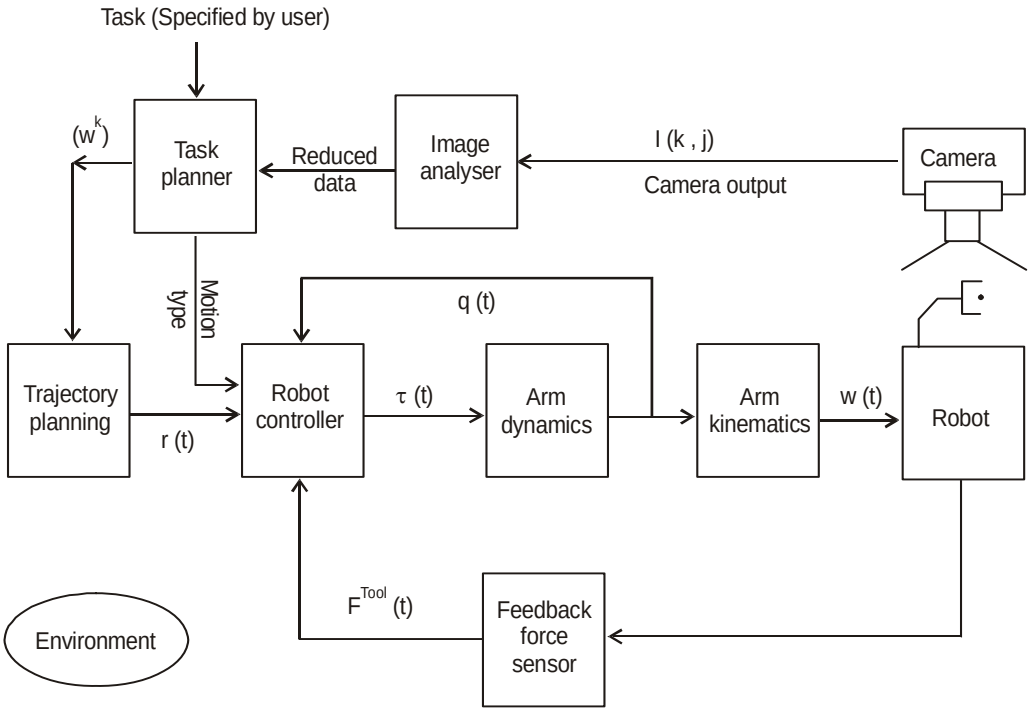

Fig. 4 A rudimentary task planner

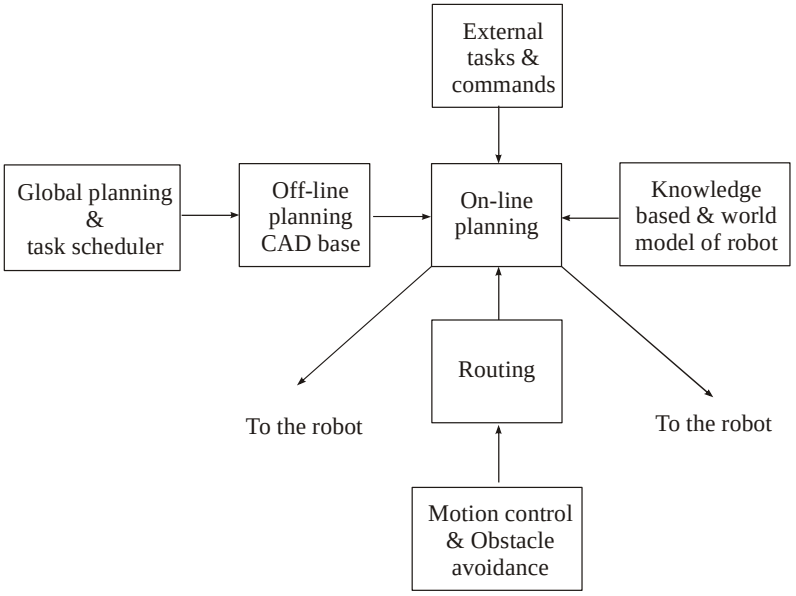

Fig. 5 Task planner design algorithm for stationary & mobile robots

V. Conclusions

A task planner was designed and also implemented in real time for the stationary robotic system. A robot was connected to the computer through the driver interfacing cards as shown in Fig. 2 and the task planner simulation software was stored in the memory of the computer. First, the task was simulated in the computer using the simulation software and then, once the task was successful, this simulated task was transformed into the reality stage using the robot.

References

[1]. Robert, J.S., "Fundamentals of Robotics : Analysis and Control", *PHI*, New Delhi., 1992.
[2]. Klafter, Thomas and Negin, "Robotic Engineering", *PHI*, New Delhi, 1990.
[3]. Fu, Gonzalez and Lee, "Robotics : Control, Sensing, Vision and Intelligence", *McGraw Hill*, Singapore, 1995.
[4]. Ranky, P. G., C. Y. Ho, "Robot Modeling, Control & Applications", *IFS Publishers, Springer*, UK., 1998.
[5]. T.C.Manjunath, "Fundamentals of Robotics", *Nandu Publishers, 5th Revised Edition*, Mumbai., 2005.
[6]. T.C.Manjunath, "Fast Track To Robotics", *Nandu Publishers, 3nd Edition*, Mumbai, 2005.
[7]. Ranky, P. G., C. Y. Ho, "Robot Modeling, Control & Applications", *IFS Publishers, Springer*, UK, 2005.
[8]. Groover, Weiss, Nagel and Odrey, Industrial Robotics, *McGraw Hill*, Singapore, 2000.
[9]. William Burns and Janet Evans, "Practical Robotics – Systems, Interfacing, Applications", *Reston Publishing Co.*, 2000.
[10]. Phillip Coiffette, "Robotics Series, Volume I to VIII, *Kogan Page*, London, UK, 1995.
[11]. Luh, C.S.G., M.W. Walker, and R.P.C. Paul, "On-line computational scheme for mechanical manipulators", *Journal of Dynamic Systems, Measurement & Control*, Vol. 102, pp. 69-76, 1998.
[12]. Mohsen Shahinpoor, "A Robotic Engineering Text Book", *Harper and Row Publishers*, UK.
[13]. Janakiraman, "Robotics and Image Processing", *Tata McGraw Hill.*
[14]. Richard A Paul, "Robotic Manipulators", *MIT press*, Cambridge.
[15]. Fairhunt, "Computer Vision for Robotic Systems", New Delhi.
[16]. Yoram Koren, "Robotics for Engineers", *McGraw Hill.*
[17]. Bernard Hodges, "Industrial Robotics", *Jaico Publishing House*, Mumbai, India.
[18]. Tsuneo Yoshikawa, "Foundations of Robotics : Analysis and Control", *PHI.*
[19]. Dr. Jain and Dr. Aggarwal, "Robotics : Principles & Practice", *Khanna Publishers*, Delhi.
[20]. Lorenzo and Siciliano, "Modeling and Control of Robotic Manipulators", *McGraw Hill.*
[21]. Dr. Amitabha Bhattacharya, "Mechanotronics of Robotics Systems".
[22]. S.R. Deb, "Industrial Robotics", *Tata McGraw Hill*, New Delhi, India.
[23]. Edward Kafrissen and Mark Stephans, "Industrial Robots and Robotics", *Prentice Hall Inc*., Virginia.
[24]. Rex Miller, "Fundamentals of Industrial Robots and Robotics, *PWS Kent Pub Co*., Boston.
[25]. Doughlas R Malcom Jr., "Robotics … An introduction", *Breton Publishing Co*., Boston.
[26]. Wesseley E Synder, "Industrial Robots : Computer Interfacing and Control", *Prentice Hall.*
[27]. Carl D Crane and Joseph Duffy, "Kinematic Analysis of Robot Manipulators", *Cambridge Press*, UK.
[28]. C Y Ho and Jen Sriwattamathamma, "Robotic Kinematics … Symbolic Automatic and Numeric Synthesis", *Alex Publishing Corp*, New Jersey.
[29]. Francis N Nagy, "Engineering Foundations of Robotics", *Andreas Siegler*, Prentice Hall.
[30]. William Burns and Janet Evans, "Practical Robotics - Systems, Interfacing, Applications", *Reston Publishing Co.*
[31]. Robert H Hoekstra, "Robotics and Automated Systems".
[32]. Lee C S G, "Robotics , Kinematics and Dynamics".
[33]. Gonzalez and Woods, "Digital Image Processing", Addison Wesseley.
[34]. Anil K Jain, "Digital Image Processing", PHI.
[35]. Joseph Engelberger, "Robotics for Practice and for Engineers", PHI, USA.
[36]. Yoshikawa T., "Analysis and Control of Robot Manipulators with Redundancy", *Proc. First Int. Symp. on Robotics Research*, Cambridge, MIT Press, pp. 735-748, 1984.
[37]. Whitney DE., "The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators", *Trans. ASM J. Dynamic Systems, Measurements and Control*, Vol. 122, pp. 303-309, 1972.

[38]. Whitney DE., "Resolved Motion Rate Control of Manipulators and Human Prostheses", *IEEE Trans. Syst. Man, Cybernetics*, Vol. MMS-10, No. 2, pp. 47-53, 1969.
[39]. Lovass Nagy V, R.J. Schilling, "Control of Kinematically Redundant Robots Using {1}-inverses", *IEEE Trans. Syst. Man, Cybernetics*, Vol. SMC-17, No. 4, pp. 644-649, 1987.
[40]. Lovass Nagy V., R J Miller and D L Powers, "An Introduction to the Application of the Simplest Matrix-Generalized Inverse in Systems Science", *IEEE Trans. Circuits and Systems*, Vol. CAS-25, No. 9, pp. 776, 1978.
[41]. Argyros A., Geordiadis P., Trahanias P., and Tsakiris D., 'Semiautonomous navigation of a robotic wheelchair', Intelligent and Robotic Systems, 34, 315–329, (2002).
[42]. Lankenau A. and Röfer T.A., 'A versatile and safe mobility assistant', Robotics and Automation Magazine, 8(1), (March 2001).
[43]. Kuipers B.J., 'The spatial semantic hierarchy, in artificial intelligence', Artificial Intelligence, 119, 191–233, (2000).
[44]. Galindo C., Fernandez J.A., and Gonzalez J., 'Improving efficiency in mobile robot task planning through world abstraction', IEEE Transaction on Robotics and Automation.
[45]. Fernandez J.A. and Gonzalez J., Multi-Hierarchical Representation of Large-Scale Space, Kluwer Academic Publishers, 2001.
[46]. Fernandez J.A. and Gonzalez J., 'Multihierarchical graph search', IEEE Transaction on Pattern Analysis and Matching, 24(1), (2002).
[47]. Hirtle S.C. and Jonides J., 'Evidence of hierarchies in cognitive maps', Memory and Cognition, 13(3), 208–217, (1985).
[48]. Yuchul Jung, Yong K. Hwang, Manjai Lee, "Case-Based Reasoning Approach toTask Planning of Home-Service Robots", ICAT 2004.
[49]. Roman Zahariev, Dimitar Karastoyanov, "A Navigation System and Task Planning in a Mobile Robot for Inspection", Bulgarian academy of sciences, Conference paper, 2004.