ISTANBUL UNIVERSITY –
JOURNAL OF ELECTRICAL & ELECTRONICS ENGINEERING
YEAR : 2004 (1161-1170)
VOLUME : 4
NUMBER : 2

REALIZATION OF REACTIVE CONTROL FOR MULTI PURPOSE MOBILE AGENTS

Selim YANNİER[1] Asif ŞABANOVİÇ[2] Ahmet ONAT[3]

Mechatronics Laboratory, Engineering and Natural Sciences Department,
Sabanci University, Orhanlı Mevkii Tuzla 34956 İstanbul, TURKEY

[1]E-mail: selimy@su.sabanciuniv.edu [2]E-mail: asif@sabanciuniv.edu [3]E-mail: onat@sabanciuniv.edu

ABSTRACT

Mobile robots are built for different purposes, have different physical size, shape, mechanics and electronics. They are required to work in real-time, realize more than one goal simultaneously, hence to communicate and cooperate with other agents. The approach proposed in this paper for mobile robot control is reactive and has layered structure that supports multi sensor perception. Potential field method is implemented for both obstacle avoidance and goal tracking. However imaginary forces of the obstacles and of the goal point are separately treated, and then resulting behaviors are fused with the help of the geometry. Proposed control is tested on simulations where different scenarios are studied. Results have confirmed the high performance of the method.
.

Keywords: *Autonomous Mobile Robot, Behavior Arbitration, Behavior Based (Reactive) Control, Multiagent System, Potential Field Method*

1. INTRODUCTION

Most of the works in the field of mobile robotics are based on one of the following assumptions: either the complete knowledge of the environment is a priori known as introduced by the operator or robot has no a priory information about the environment [1-3].

First method is “model based” and generally referred as “deliberative control” [2]. Requirement of a complete model of the environment is the main difficulty in those systems. Other drawbacks of this approach are the high computational power and large memory requirements [1, 2, 4]. Moreover, they do not effectively resolve navigation problems in real-world applications where multiple moving obstacles are involved [5].

Second approach considers the task as a combination of more elementary tasks called “behaviors” [4, 6, 7]. Programming the execution of a given task then reduces to finding the proper combination of those behaviors to produce the desired task. This method is “sensor based” and referred as “reactive control” or “behavior based control” [1, 2].

Many results on behavior-based control of mobile robots [4, 6, 7] with variety of obstacle avoidance methods [5, 8] are already published. Tunstel used fuzzy logic based controllers in his

Received Date: 23.08.2003
Accepted Date: 15.06.2004

behavior based mapping robot [9]. A similar study is carried by Tsourveloudis also [10]. Luo and Chen used behavior based mobile robot to avoid disturbances of the Internet latency in remote control [11, 12]. Parker [13] has modified behavioral controls for multi-robot cases. Arkin [14] extended behavioral control architecture for multi-robot control. Eustace [15] created "Behavioral Synthesis Model" designed to facilitate cooperation and coordination between multiple robotic devices for execution of complex tasks. Fontan and Mataric demonstrated the application of the distributed behavior based approach to generating a multi robot controller [16].

There are already several implementations of primitive behaviors using variety of methods showing high performance when executed for one task at the time only. However, once multiple goal realization, such as avoiding obstacle while reaching a target point, comes into picture then action selection is the key issue. For action selection, Brooks used subsumption architecture; each layer runs in parallel, however, the output of only one is executed in a specific time [4]. Although this configuration works well in less crowded areas such as laboratory test beds, in a real world application results were not so successful. Consequently, better action selection methods were needed [2].

Many researchers suggested and applied fuzzy logic based controllers [9, 17, 18]. The advantage of fuzzy logic is that potentially conflicting functions can be fused in a natural and smooth way, so that a reasonable decision can be made to serve both functions. Mochiada proposed an "Emotional Mechanism" similar to the human emotional mechanism as a solution [19].

The development of satisfactory control method for an autonomous mobile robot that can be part of a multiagent system is still an open problem. For such a system, one can identify a number of requirements,

- *Multigoal support:* control of a mobile robot must find the way to select the action that serves a maximum number of goals at the same time.
- *Robustness:* in the case of failures or erroneous readings of the sensors, the robot must still show meaningful behavior within limits.
- *Platform independence:* it should be applicable to mobile robots with different physical size, shape, mechanics and electronics.
- *Cooperative:* a mobile robot control must be open for additional controls that will guide the robot to be a part of a multiagent system.

The goal of this work is to present a new structure for the mobile robot motion control system, capable of being a building block of an intelligent agent.

The rest of the paper is organized as follows. Section 2 describes the plant. The whole control that is designed is presented in Section 3. Section 4 presents the simulation results of the proposed method are presented. Conclusions and areas for future research are presented in Section 5.

2. Plant

The plant consists of two main entities: agents and obstacles.

2.1. Simplified Model of Agents

Sample mobile agent is differential drive type, nonholonomic robot generally referred as "wheel set" as shown in Figure 1. Kinematics of such a robot can easily be determined assuming no slip at tires [20],

$$\begin{aligned} \dot{x} &= v \cdot \cos\phi \\ \dot{y} &= v \cdot \sin\phi \\ \dot{\phi} &= \omega \end{aligned} \quad , \quad \begin{aligned} v &= (v_R + v_L)/2 \\ \omega &= (v_R - v_L)/L \end{aligned} \qquad (1)$$

where $q = (x, y, \phi) \in \Re^3$ is the state of the robot represented by position and the orientation in world coordinate frame (x_w, y_w), L denotes the length of the axis joining driven wheels and v is the velocity of the center of the two driving wheels. Variables that should be controlled are right and left wheel's linear velocities, v_R and v_L respectively, which may easily be translated into the translational and rotational velocity variables $u = (v, \omega) \in \Re^2$ for convenience [20].

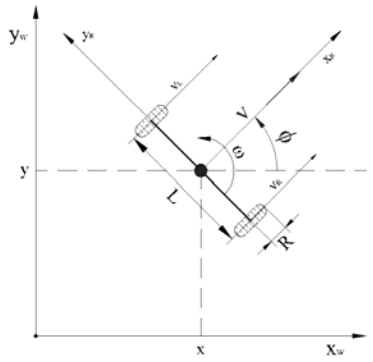

Figure 1. Wheel set is used as sample physical agent.

2.1.1. Sensors of the Agent

Selected agents have two major types of sensors. First type is for internal usage and is necessary for feedback control, such as encoders to detect the position and/or velocity of the driving motors. Second type sensors are for detecting the environmental states such the place of the obstacles by ultrasonic distance sensors.

2.2. Obstacles

For practical reasons we are referring to all physical objects present in the environment (including other agents) as obstacles. Obstacles are entities that are either preventing the agent to move or limiting its actions.

3. PROPOSED SOLUTİON, SYSTEM LAYER DESIGN

Proposed control is a layered structure formed out of two types of layers: parallel and serial as shown in Figure 2. Parallel layers are "competence layers" that are performing their own tasks independently and most producing an output in the form of "desired" velocity and orientation change. Serial layers on the other hand are the connections of the parallel layers to the hardware. Details of each layer are presented in the following sections.

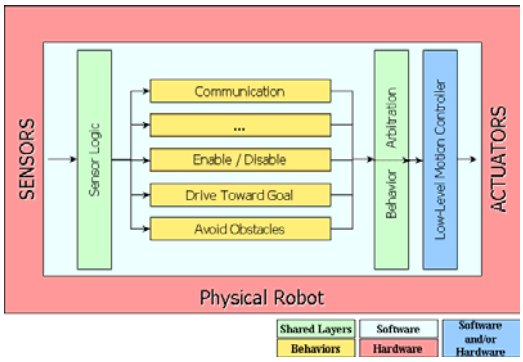

Figure 2. Structure of the proposed solution.

3.1. Layer 0: Low-Level Motion Controller

Layer 0 represents all hardware such as the body of the robot, actuators, drivers and speed/position controllers, wheels, sensors etc. Moreover, this is the layer where the reference velocity and direction information from higher levels are converted to reference wheel velocities in so-called low-level motion controller. Finally, the output of this controller constitutes speed references for wheel velocity controllers.

First, using actual position of the robot (x, y) together with the reference velocity v_{ref} and orientation ϕ_{ref}, reference position of the robot can be obtained,

$$\begin{aligned} \dot{x}_{ref} &= v_{ref} \cdot \cos\phi_{ref} \\ \dot{y}_{ref} &= v_{ref} \cdot \sin\phi_{ref} \end{aligned} \tag{2}$$

Those two references can be combined,

$$r_{ref} = \sqrt{x_{ref}^2 + y_{ref}^2} \tag{3}$$

where r_{ref} corresponds to the distance from the origin of the world coordinate frame to the robot's reference position. Obviously, the control should be selected such that position errors $e_x = x_{ref} - x$ and $e_y = y_{ref} - y$ can be kept under certain threshold. Projection of those two errors on to the velocity and steering direction axis (denoted with subscript r and ϕ respectively) can be found.

$$\begin{aligned} e_r &= e_x \cdot \cos\phi + e_y \cdot \sin\phi \\ e_\phi &= -e_x \cdot \sin\phi + e_y \cdot \cos\phi \end{aligned} \tag{4}$$

We can then calculate "corrected" values for the reference values and corrected errors as

$$\begin{aligned} r_{ref}^{corr} &= r_{ref} + e_r \\ \phi_{ref}^{corr} &= \phi_{ref} + e_\phi \end{aligned} \quad \Rightarrow \quad \begin{aligned} e_r^{corr} &= r_{ref}^{corr} - r = \sigma_r \\ e_\phi^{corr} &= \phi_{ref}^{corr} - \phi = \sigma_\phi \end{aligned} \tag{5}$$

Choosing

$$u_1 = v_R + v_L \qquad \text{and} \qquad u_2 = v_R - v_L \tag{6}$$

as controls and using eq-3, eq-1 becomes;

$$\begin{aligned} \dot{r} &= u_1/2 \\ \dot{\phi} &= u_2/L \end{aligned} \tag{7}$$

Note that u_1 is proportional to v while u_2 is proportional to $\dot{\phi} = \omega$.

The control should be chosen such that components of the positive definite Lyapunov function candidate $\gamma = \sigma^T\sigma/2 \geq 0$ satisfy Lyapunov stability criteria. Since both equations are independent, we can use componentwise control, where components of the error vector are separately controlled to tend to zero. Separating γ to its components;

$$\begin{aligned} \dot{\gamma}_i &= \sigma_i\dot{\sigma}_i = -D_i \cdot \sigma_i^2 \\ \sigma_i\left(\dot{\sigma}_i + D_i \cdot \sigma_i\right) &= 0 \end{aligned} \quad , \qquad i = r, \phi \tag{8}$$

where, $\gamma_i \geq 0$ and $\dot{\gamma}_i \leq 0$, for $i = r, \phi$ and for some constant $D > 0$. In the above equation, either σ_i or $\left(\dot{\sigma}_i + D_i \cdot \sigma_i\right)$ is zero. if $\left(\dot{\sigma}_i + D_i \cdot \sigma_i\right)$ is zero for $\sigma_i \neq 0$, then obviously σ_i will tend to zero.

Solving above equation for discrete time systems where small computational delays are neglected we obtain [20];

$$\begin{aligned} u_1^k &= u_1^{k-1} + \frac{1}{dt}\left(\left(1 + dt \cdot D_r\right) \cdot \sigma_r^k - \sigma_r^{k-1}\right) \\ u_2^k &= u_2^{k-1} + \frac{1}{dt}\left(\left(1 + dt \cdot D_\phi\right) \cdot \sigma_\phi^k - \sigma_\phi^{k-1}\right) \end{aligned} \tag{9}$$

where dt stands for discrete time interval, and k denotes the k^{th} time interval. Clearly, u^k belongs to the current time interval while u^{k-1} represents the past value.

Finally, actual references for the right and left wheel velocities that will be used by servo controllers are found as,

$$\begin{aligned} v_R^{ref} &= \left(u_1 + u_2\right)/2 \\ v_L^{ref} &= \left(u_2 - u_1\right)/2 \end{aligned} \tag{10}$$

3.2. Layer 1: Obstacle Avoidance (OA)

For obstacle avoidance, potential field method is used [21, 22]. This method is particularly attractive because of its simplicity and compatibility with different type of sensors.

The basic concept of the potential field method is to fill the robot's environment with an artificial potential field created by imaginary forces of the form,

$$\vec{F}_{obs} = -\frac{A}{d^2} \cdot \hat{r} \tag{11}$$

where A is a constant scaling factor, d is the distance between obstacle and agent from sensor readings, and $\hat{r}$ is the direction from the agent to the obstacle. By the way, obstacles repel the robot. Moreover, the inverse proportionality ensures significant increase in force magnitude when the agent is close to obstacles, which cause stronger reaction to avoid collisions.

Since the force is the negative gradient of the field ($\vec{F}_{obs} = -\vec{\nabla}U(d)$), the agent can calculate the potential field created by sensed obstacles at any point in the space (Figure 3). Nevertheless, the agent might not be able to detect every obstacle present in the environment since this depends on number, orientation and range of the sensors. Therefore, the experienced potential field might be slightly different from the expected one.

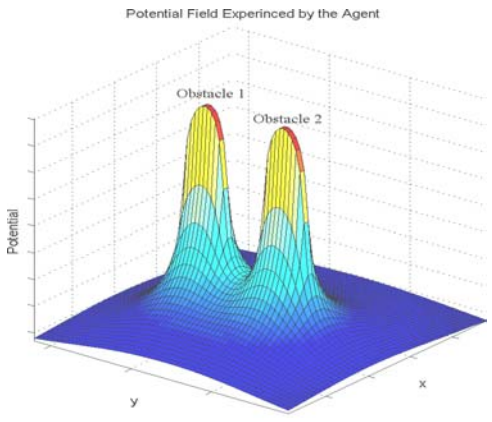

Figure 3. Potential field created by two obstacles.

In many applications, the repulsive force directly influences the motions of the robot by the use of classical Newtonian law $\vec{F} = m\vec{a}$ where $\vec{F}$ is the net force assumed to move the robot, m is the mass (more generally used as a scaling factor) and $\vec{a}$ is the corresponding robot acceleration vector [22]. However, $\vec{F}$ is always in the decreasing potential direction, and therefore robot is bounded to move opposite direction of the encountered obstacle regardless of the position of the goal point. A better approach is to find the way to make robot follow the obstacle boundary so that it can go around it to reach other side where probably goal point is located.

First, we decompose $\vec{F}_{obs}$ into its components: one along velocity direction of the agent $\vec{F}_r$ and other in the direction perpendicular to it, $\vec{F}_\phi$.

$$\begin{aligned} F_r &= \left\|\vec{F}_{obs}\right\| \cdot \cos\theta \\ F_\phi &= \left\|\vec{F}_{obs}\right\| \cdot \sin\theta \end{aligned}, \qquad \begin{aligned} &\theta = \phi - \theta_{obs} \\ &-\pi \le \theta \le \pi \end{aligned} \tag{12}$$

where θ_{obs} is the orientation of $-\vec{F}_{obs}$ (from robot to the obstacle) in world coordinate frame.

For a safe travel, the agent must be reoriented to keep $\vec{F}_r$, the force along the heading direction, minimum or generally zero, $F_r^{ref} = 0$. The rate of change of those components is,

$$\begin{aligned} \dot{F}_r &= -\left\|\vec{F}_{obs}\right\| \cdot \dot{\theta} \cdot \sin\theta = -F_{obs} \cdot \left(\dot{\phi} - \dot{\theta}_{obs}\right) \cdot \sin\theta \\ \dot{F}_\phi &= \left\|\vec{F}_{obs}\right\| \cdot \dot{\theta} \cdot \cos\theta = F_{obs} \cdot \left(\dot{\phi} - \dot{\theta}_{obs}\right) \cdot \cos\theta \end{aligned} \tag{13}$$

From here, one can conclude that control of both F_r and F_ϕ is feasible by changing orientation of the robot. This fact may be used for establishing structure in which the obstacle avoidance layer will be used to change orientation of the agent thus influencing "reference motion" instead of interfering with low-level motion control. This way the motion control loop is embedded in the obstacle avoidance loop.

By representing obstacle avoidance loop as two dimensional system,

$$\dot{F}_r = u_r^{OA} \quad , \quad \dot{F}_\phi = u_\phi^{OA} \tag{14}$$

one can design an OA controller following the same steps as in motion control. We can now define errors to be minimized,

$$\begin{aligned} e_r^{OA} &= F_r^{ref} - F_r \\ e_\phi^{OA} &= F_\phi^{ref} - F_\phi \end{aligned} \tag{15}$$

Using Lyapunov Function candidate $\gamma = \mathbf{e}_{OA}{}^T \mathbf{e}_{OA} / 2 \ge 0$ and procedure described in section 0 we obtain

$$\begin{aligned} u_r^{OA,k} &= u_r^{OA,k-1} + \frac{1}{dt}\left(\left(1 + dt \cdot D_r^{OA}\right) \cdot e_r^{OA,k} - e_r^{OA,k-1}\right) \\ u_\phi^{OA,k} &= u_\phi^{OA,k-1} + \frac{1}{dt}\left(\left(1 + dt \cdot D_\phi^{OA}\right) \cdot e_\phi^{OA,k} - e_\phi^{OA,k-1}\right) \end{aligned} \tag{16}$$

Using eq-14 and eq-16 together

$$\frac{u_r^{OA}}{u_\phi^{OA}} = \frac{-\sin\theta^{OA}}{\cos\theta^{OA}} \Rightarrow \theta^{OA} = \tan^{-1}\left(-\frac{u_r^{OA}}{u_\phi^{OA}}\right) \tag{17}$$

where θ^{OA} is the reference orientation created by obstacle avoidance layer for a collision free path. All values in the above equation are for the present time. For practical reasons, the output of this layer is converted to "the desired change in the orientation"

$$\Delta\phi^{OA} = \phi - \theta^{OA} \tag{18}$$

before sent to the next layer.

As one can see the obstacle avoidance controller has the same structure as motion controller. They are structurally connected in such a way that OA layer modifies the behavior "references" of the motion control level.

3.3. Layer 2: Drive Toward Goal Point (DTG)

Potential field method is not only used for obstacle avoidance purposes but also for goal tracking. In potential field method, the agent is forced move toward the region of the space where the potential created by obstacles is minimum. However, this does not ensure the robot to reach a specific point namely the goal point. However, if in addition to the imaginary repulsive forces (eq-11), an attractive force toward the goal point is added, minima of the potential field will occur at that point (Figure 4). This force has generally the form,

$$\vec{F}_{atr} = B \cdot d^2 \cdot \hat{r} \tag{19}$$

where B is a constant scaling factor, d is the distance to the goal point and $\hat{r}$ is the direction from the agent to that point.
In most applications, this force is summed with the repulsive forces and the resultant force is used to navigate the robot according to the classical Newtonian law $\vec{F} = m\vec{a}$. However, in case of conflicts like an obstacle between the moving agent and the goal point, robot will be forced to move to opposite side to avoid that obstacle if no additional precaution is taken. Moreover, many local minimums may appear in the environment, especially close to the passages like door openings etc, and in general, robots are stuck at those points.

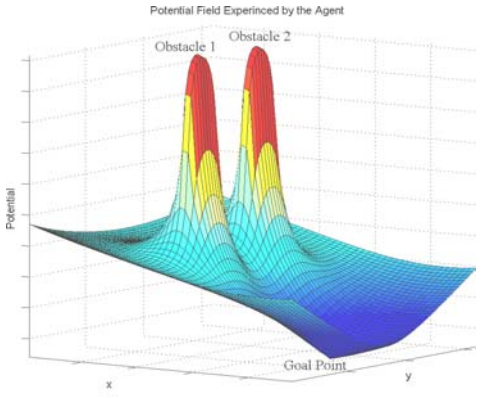

Figure 4. Potential field created by two obstacles and a goal point.

In our application, we deliberately selected to treat those forces separately in two different layers, in order to avoid such problems. The attractive force $\vec{F}_{atr}$ is first decomposed into its components: one along velocity direction of the agent $\vec{G}_r$ and other in the direction perpendicular to it $\vec{G}_\phi$.

$$\begin{aligned} G_r &= \left\|\vec{F}_{atr}\right\| \cdot \cos\theta' \\ G_\phi &= \left\|\vec{F}_{atr}\right\| \cdot \sin\theta' \end{aligned} \quad , \quad \begin{aligned} \theta' &= \phi - \theta_{atr} \\ -\pi &\le \theta' \le \pi \end{aligned} \tag{20}$$

To obtain an orientation toward the goal point the force along the heading direction must be maximized $G_r^{ref} = \left\|\vec{F}_{atr}\right\|$, while the other component is forced to be minimum $G_\phi^{ref} = 0$.

By following the same reasoning in OA we can easily find the rate of change of the goal forces as,

$$\begin{aligned} \dot{G}_r &= -\left\|\vec{F}_{atr}\right\| \cdot \dot{\theta} \cdot \sin\theta' = u_r^{DTG} \\ \dot{G}_\phi &= \left\|\vec{F}_{atr}\right\| \cdot \dot{\theta} \cdot \cos\theta' = u_\phi^{DTG} \end{aligned} \tag{21}$$

we obtain our control variables u_r^{DTG} and u_ϕ^{DTG}. The errors to be minimized are,

$$\begin{aligned} e_r^{DTG} &= G_r^{ref} - G_r \\ e_\phi^{DTG} &= G_\phi^{ref} - G_\phi \end{aligned} \tag{22}$$

Using Lyapunov Function candidate $\gamma = \mathbf{e}_{DTG}^{\ T}\mathbf{e}_{DTG}/2 \ge 0$ and procedure described in section 0 we obtain

$$\begin{aligned} u_r^{DTG,k} &= u_r^{DTG,k-1} \\ &\quad + \frac{1}{dt}\left(\left(1 + dt \cdot D_r^{DTG}\right) \cdot e_r^{DTG,k} - e_r^{DTG,k-1}\right) \\ u_\phi^{DTG,k} &= u_\phi^{DTG,k-1} \\ &\quad + \frac{1}{dt}\left(\left(1 + dt \cdot D_\phi^{DTG}\right) \cdot e_\phi^{DTG,k} - e_\phi^{DTG,k-1}\right) \end{aligned} \tag{23}$$

Using eq-21 and 23 together,

$$\frac{u_r^{DTG}}{u_\phi^{DTG}} = \frac{-\sin\theta^{DTG}}{\cos\theta^{DTG}} \Rightarrow \theta^{DTG} = \tan^{-1}\left(-\frac{u_r^{DTG}}{u_\phi^{DTG}}\right) \tag{24}$$

where θ^{DTG} is the reference orientation created by drive toward goal layer to move the robot toward the requested location. For practical reasons, the output of this layer is converted to "the desired change in the orientation"

$$\Delta\phi^{DTG} = \phi - \theta^{DTG} \tag{25}$$

before being sent to the next layer.

3.4. Behavior Arbitration

Unless disabled by a higher layer, DTG (Layer 2) is working and producing an output, $\Delta\phi^{DTG}$. In addition, once the robot senses an obstacle, OA (Layer 1) will produce another output, $\Delta\phi^{OA}$, that is most probably in conflict with the other one.

Agent must avoid obstacles while driving toward the goal point. Therefore, $\Delta\phi^{DTG}$ and $\Delta\phi^{OA}$ must be combined such that both request are partially fulfilled. For this purpose, serially placed behavior arbitration layer calculating the weighted sum of $\Delta\phi^{DTG}$ and $\Delta\phi^{OA}$ to transmit the velocity and orientation references, to the low-level motion controller, is proposed.

In this process, weights are not constant and are calculated from geometrical relationships. Assuming that the agent is moving toward the goal point, while avoiding an obstacle in midway as shown in Figure 5 below.

Observing the situation, we can see that when the angle between vectors $\vec{F}_{obs}$ and $\vec{v}$ is close to π, obstacle avoidance must gain importance. On the other hand, if this angle is close to $\pi/2$ then obstacle is close to the either side of the robot and therefore the collision has low probability. In this case, the importance of DTG must be increased. Mathematically this can be shown as,

$$\phi^{ref} = \phi + A^2 \cdot \Delta\phi^{OA} + B^2 \cdot \Delta\phi^{DTG} \tag{26}$$

where ϕ is the actual orientation of the robot and A and B are the complimentary constants $A + B = 1$ that represents the weights in the summation. They are both used as square to increase smoothness in the reference orientation ϕ^{ref} and are derived using θ: the angle between velocity $\vec{v}$ of the robot and repulsive force $\vec{F}$:

$$A = 1 - B = \begin{cases} 1\,(\max) & for \quad \theta = \pi \\ \vdots & \\ 0\,(\min) & for \quad \theta \le \pi/2 \end{cases} \tag{27}$$

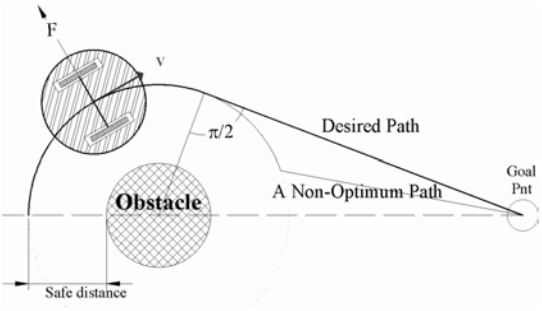

Figure 5: Optimum and non-optimum path example for an agent while avoiding an obstacle.

In this layer, velocity reference is not changed. However, a deceleration when an obstacle is detected and acceleration when the path is free could also be added in this layer.

The output of this layer is the reference velocity v^{ref} and orientation ϕ^{ref} that is sent to the low-level motion controller where the motor velocities are calculated and controlled accordingly.

3.5.Layer 3: Enabling and Disabling Features

In different applications during general use, because of possible restrictions, a specific behavior may be required to be disabled, as in the example of a mobile robot that needs to stop and charge it-self at the station.

The third layer of the proposed algorithm is responsible of the enabling and disabling of the three features:

- **Drive Enable (Move or Stop):** The desired task may require being stationary at a given point as in the case of a carrier robot that is close enough to its load to grasp it.
- **Drive Toward Goal Enable (DTG Enable):** It may also be necessary to disable DTG layer. For example, if the agent is stuck among obstacles and cannot move simply because of the configuration, it may be useful to temporarily disregard the goal.
- **Obstacle Avoidance Enable:** There are cases where even obstacle avoidance must be disabled. A forklift approaching to a box to hold it must disable this layer since otherwise it will be forced to move away by the commands of OA layer. In such an application disabling sensors is not a suitable solution since this control shares all available resource to the entire layered structure.

3.6.Layer 4: Longer Term Memory Layers

Most of the researchers aim to create mobile robots that can work in hazardous environments such as a deep sea, nuclear plants and polluted areas where humans may not survive. Generally, the robot must record some data such as temperature, nuclear radiation, altitude etc. and carry it to the base station where further analysis is done. Similarly a carrier robot that is working in a factory, must keep a log of what is transported.

Such kind of data logging work should be realized in this layer. However, processing of this data or actions related to the situation must be applied in the upper layer.

3.7. Layer 5: User Defined Layers

In real robotics applications, most of the time there is an external hardware, which must be controlled. A mobile robot may have a robot arm mounted on top of it together with a suitable end-effector to realize tasks such as painting a wall. In automated carrier, the robot must have hardware to grasp, lift, move and leave objects.
The control related to this hardware or any modification to the existing control layers (such as changing reference values of the force control layers) should be done from this layer. Control placed in this layer, as all other layers, have access to the sensor data, and to all lower level blocks. If this layer will generate a modification request in reference velocity and orientation of the low-level motion controller this must be done through behavior arbitration layer that will be modified accordingly.

3.8. Layer 6: Communication

Communication is the only link between the agents and the user. High level command such as "move to (x, y)", "start/stop execution of tasks" and low level commands such as "disable DTG", "open gripper" are sent to the agent using this link.
Similarly, collected data by the agent is transmitted to the user and other agent by this link. Moreover, communication can safely be used in multi-robot collaboration where small time delays due to the transmission time are not important.
This layer must be at the top layer from where it can reach and modify all other layers. Any suitable communication method can be applied.

4. SIMULATIONS AND RESULTS

Proposed control for mobile robots is tested on the developed simulation tool written in C++ programming language. This simulation code, consists of a core which is a collection of routines and experiments that are defining the experimental setup and then calling routines in the necessary order. Results are shown using a simple GUI.

4.1. Stationary Obstacles

In this experiment, only one agent is placed in the environment together with four stationary obstacles. Agent is told to traverse toward the other side of the environment (see Figure 6). What is expected from the agent is a smooth and safe navigation through obstacles. Agent must direct itself toward the goal point unless an obstacle is sensed. In such a case, the direction of navigation must be changed to go around the obstacle at a safe distance. This is exactly what is observed in the experiment.
Furthermore, in this figure, we also see clearly the work done by the behavior arbitration: when the agent gets close to the obstacle 1, it was moving directly to the obstacle. The force control algorithm influenced the robot to change its direction such that, the robot started to move around the obstacle 1. At the point shown with an arrow in Figure 6, obstacle 1 is not between the agent and the goal point anymore. Consequently, the behavior arbitration inhibits the output from the obstacle avoidance layer until the agent reaches to the sensibility range of the obstacle 2. A similar behavior is observed with the two other obstacles.

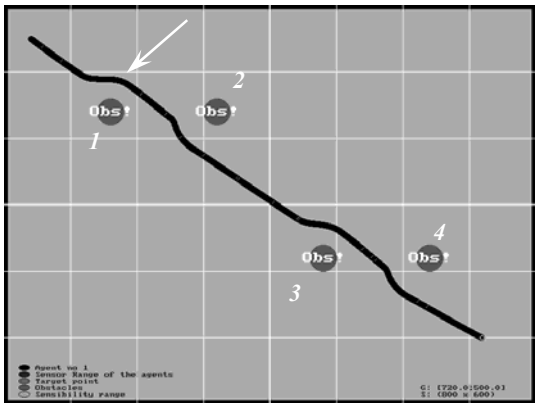

Figure 6. Avoidance of stationary obstacles.

4.2. Moving Obstacles

In this experiment, we tested the reaction of the agent to the moving obstacles. As shown in Figure 7, an agent is placed at the point S and told to move to the point T. Four other agents, with obstacle avoidance layer disabled, are also placed to the environment (MO1, MO2, MO3 and MO4).

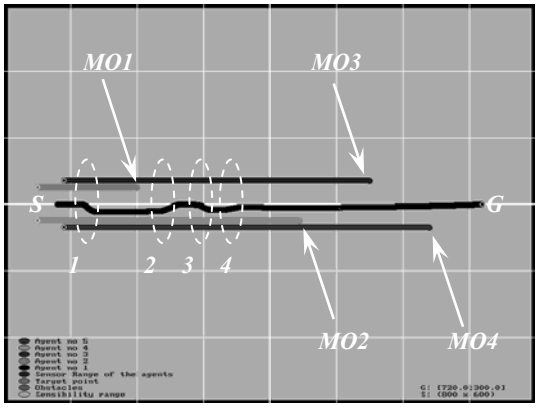

Figure 7. Avoidance of moving obstacles.

During the experiment, agent confronted four moving obstacles, simulating humans or rolling balls, one by one. First confrontation happened with MO1 (see circled area marked as 1 in Figure 7). The agent reacted quickly to avoid the obstacle. As expected, this reaction was fast since both the force and its derivative is used in control. When the path was clear to the robot, it reoriented it-self toward the target point T, until next confrontation. Similar behavior is observed for MO2, MO3 and MO4 confrontations. We see clearly that the agent moves naturally and safely in the area where it encounters moving obstacles continuously.

5. CONCLUSION

In this work, we suggested a new approach for realization of reactive control of mobile robots. This new realization divides the control into layers. Each layer has its own task and goals to be accomplished. Outputs from each layer are collected in behavior arbitration. In the whole control, behavior arbitration is the only part that can directly influence the motion of the robot, except some possible user applications such as an emergency stop command.

The proposed approach for realization supports multi goals. Reaching to a specific point while avoiding obstacles is a simple multi goal example for a mobile robot. Those two basic goals for a mobile robot are already in the control, and working in harmony. Further goals can easily be defined in the appropriate layer of control. This way, the additions of the new layers to the mobile robot control will augment richness of the behaviors observed and with correct implementations will increase the performance observed.

Proposed control is tested on simulations, and different scenarios are studied. Especially, the cases that are problematic to many other approaches are investigated. Some of the results are shown in the previous chapter. The simulation results confirmed the high performance of the method. Moreover, same results show that some of the drawbacks coming from the nature of the applied control are avoided. Furthermore, from these results, we can conclude that the proposed control is a potential alternative for mobile robots control operating in dynamic environments and/or as an agent in multiagent system.

REFERENCES

[1] R. C. Arkin, *Behavior-Based Robotics*. Cambridge, Mass.: MIT Press, 1998.

[2] J. Ferber, *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Harlow, Eng.: Addison-Wesley, 1999.

[3] W. L. Xu and S. K. Tso, "Sensor-Based Fuzzy Reactive Navigation of a Mobile Robot through Local Target Switching" *IEEE Trans. on Systems, Man and Cybernetics, Part C*, vol. 29, pp. 451-459, 1999.

[4] R. A. Brooks, "A Robust Layered Control System for a Mobile Robot" MIT Artificial Intelligence Laboratory, Massachusetts A.I. Memo 864, Sep. 1985.

[5] K.-T. Song and C. C. Chang, "Reactive Navigation in Dynamic Environment Using a Multisensor Predictor" *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 29, pp. 870-880, 1999.

[6] R. A. Brooks, "A Robot That Walks; Emergent Behaviors from a Carefully Evolved Network" MIT Artificial Intelligence Laboratory A.I. Memo 1091, Feb. 1989.

[7] R. A. Brooks, *Cambrian Intelligence: The Early History of the New A.I.* Cambridge, Mass.: MIT Press, 1999.

[8] A. Steinhage and R. Schoner, "The Dynamic Approach to Autonomous Robot Navigation" presented at IEEE International Symposium on Industrial Electronics, ISIE' 97, 1997.

[9] E. Tunstel and M. Jamshidi, "Fuzzy Logic and Behavior Control Strategy for Autonomous Mobile Robot Mapping" presented at IEEE World Congress on Computational Intelligence, Third IEEE Conference on Fuzzy Systems, 1994.

[10] N. C. Tsourveloudis, K. P. Valavanis, and T. Hebert, "Autonomous Vehicle Navigation Utilizing Electrostatic Potential Fields and Fuzzy Logic" *IEEE Trans. on Robotics and Automation*, vol. 17, pp. 490-497, 2001.

[11] R. C. Luo and T. M. Chen, "Development of a Multi-Behavior Based Mobile Robot for Remote Supervisory Control through the Internet" *IEEE/ASME Trans. on Mechatronics*, vol. 5, pp. 376 -385, 2000.

[12] T. M. Chen and R. C. Luo, "Development and Integration of Multiple Behaviors for Autonomous Mobile Robot Navigation" presented at Proc. of the 24th Annual Conf. of the IEEE Industrial Electronics Society, IECON '98, 1998.

[13] L. E. Parker, "A Performance-Based Architecture for Heterogeneous, Situated Agent Cooperation" presented at AAAI Workshop on Cooperation Among Heterogeneous Intelligent Systems, 1992.

[14] R. C. Arkin, T. Balch, and E. Nitz, "Communication of Behavioral State in Multi-Agent Retrieval Tasks" presented at IEEE Int. Conf. on Robotics and Automation, 1993.

[15] D. Eustace, D. P. Barnes, and J. O. Gray, "A Behavior Synthesis Architecture for Co-Operant Mobile Robot Control" presented at International Conference on Control, Control '94, 1994.

[16] M. Schneider-Fontan and M. J. Mataric, "Territorial Multi-Robot Task Division" *IEEE Trans. on Robotics and Automation*, vol. 14, pp. 815-822, 1998.

[17] C. Ma, W. Li, and L. Liu, "Mobile Robot Motion by Integration of Low- Level Behavior Control and High Level Global Planning" presented at IEEE International Conference on Systems, Man and Cybernetics, 1996.

[18] S. S. Ge and Y. J. Cui, "New Potential Functions for Mobile Robot Path Planning" *IEEE Trans. on Robotics and Automation*, vol. 16, pp. 615-620, 2000.

[19] T. Mochiada, A. Ishiguro, T. Aoki, and Y. Uchikawa, "Behavior Arbitration for Autonomous Mobile Robots Using Emotion Mechanisms" presented at IEEE/RSJ International Conference on Intelligent Robots and Systems 95, 'Human Robot Interaction and Cooperative Robots', 1995.

[20] S. Yannier, "Realization of Reactive Control for Multi Purpose Mobile Agents," in *Electronics Engineering and Computer Sciences*. Istanbul: Sabanci University, 2002, pp. 107.

[21] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots" presented at IEEE International Conference on Robotics Automation, St. Louis, MO, 1985.

[22] J. Borenstein and K. Y., "The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots" *IEEE Transactions on Robotics & Automation*, vol. 7, pp. 278-287, 1991.