International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:8, No:5, 2014

Deadline Missing Prediction for Mobile Robots through the Use of Historical Data

Edwaldo R. B. Monteiro, Patricia D. M. Plentz, Edson R. De Pieri

Abstract—Mobile robotics is gaining an increasingly important role in modern society. Several potentially dangerous or laborious tasks for human are assigned to mobile robots, which are increasingly capable. Many of these tasks need to be performed within a specified period, i.e, meet a deadline. Missing the deadline can result in financial and/or material losses. Mechanisms for predicting the missing of deadlines are fundamental because corrective actions can be taken to avoid or minimize the losses resulting from missing the deadline. In this work we propose a simple but reliable deadline missing prediction mechanism for mobile robots through the use of historical data and we use the Pioneer 3-DX robot for experiments and simulations, one of the most popular robots in academia.

Keywords—Deadline missing, historical data, mobile robots, prediction mechanism.

I. Introduction

IN today's society there is an increasing need to perform tasks very efficiently and accurately. Many of these tasks must be performed in hazardous and remote locations, such as the sea or space. This need has stimulated the creation of devices, called robots, which perform these tasks without risk to human life. A robot is an electromechanical agent able to perform various tasks with human assistance or independently. In recent years, we have noticed a large growth in the potential of these robots, gaining an increasingly important role in our daily lives. They are widespread in the industry, where they perform difficult and elaborate tasks with great precision and speed. In this case, they are called industrial robots.

Mobile robotics has been the focus of much study and has become an area of great importance and interest to researchers and engineers. Mobile robots are robots that have the ability to move in a given environment, not limited to a fixed location like the industrial robot manipulators. Autonomous mobile robots are able to perform tasks without assistance or human interaction and have various applications in the military, industry, medicine, entertainment, research and so on. One of the main aspects to consider in mobile robotics is the environment where the robot will move around. To navigate in a known environment, the robot typically uses a map of this environment. The environment may contain unknown objects, prohibited areas and moving obstacles. The robot uses the map information, along with data obtained from its sensors to navigate safely. Another important aspect is the path planning, which allows a safe path to be defined from the initial position of the robot to the final destination, avoiding known and unknown obstacles during the navigation.

Edwaldo R. B. Monteiro and Patricia D. M. Plentz are with the *Programa de Pós-Graduação em Ciência da Computação* (PPGCC) of the Federal University of Santa Catarina (UFSC), Florianóplis SC, Brazil (e-mail: dymont inf.ufsc.br and patricia.plentz ufsc.br).

Edson R. De Pieri is with the *Automation and Systems Departament* (DAS) of the Federal University of Santa Catarina (UFSC), Florianóplis SC, Brazil (e-mail: edson das.ufsc.br).

Real-time systems are computational systems where the execution time of a given task is crucial. These systems demand logically correct results but also require that their responses to the environment should be given in a timely manner so that the system does not come in an inconsistent or invalid state. The system must be able to perform the task within a given deadline or inform the task cannot be executed, if its completion is undesirable after the deadline missing. Thus, the correctness of the system depends not only on the results of the computation, but also on the time at which these results are produced. Commonly, real-time systems are wrongly associated with systems that produce fast results. However, what defines a system as a real-time is essentially the time constraint, e.g., it must meet deadlines. The real-time systems can be classified as *hard* or *soft* depending on the consequences of missing a deadline. In Hard Real-Time Systems, deadline misses are not tolerable and can result in irreversible consequences. On the other hand, in Soft Real-Time Systems, time remains crucial for the application, but the system can miss some deadlines. However, eventually performance will degrade if too many deadlines are missed.

Mobile robots are an example of a real-time system. They consist of a set of subsystems - actuators, sensors and a software subsystem. These subsystems work together so that the robot can meet the deadlines of the low-level tasks, such as detecting the presence of an unknown obstacle and avoiding a collision. It is essential that these subsystems work properly so the robot can perform high-level tasks, such as moving from an initial position to a final position. Disregarding possible failures in the subsystems mentioned above and considering they can perform low-level tasks (reading sensors, motor drive, etc.) within the given deadline, we can also apply timing constraints in the robot high-level tasks. These tasks, in addition to being logically correct, they must meet a certain deadline. The ability to predict the missing of a deadline is important in various scenarios. For example, in an automated oil platform, if a task is not completed within the given deadline, it can cause delays and serious material and financial losses. In this sense, it is important to develop reliable mechanisms that can predict a deadline missing while the robot is performing tasks, so that corrective actions can be taken to avoid or minimize the losses resulting from missing the deadline.

This paper proposes a simple but reliable deadline missing forecast mechanism for mobile robots performing high-level tasks through the use of data from previous executions of

similar tasks. These tasks consist in moving the robot from an initial position to another position and back to the original position. While performing these tasks, the robot can carry loads, provide assistance to other robots, etc, within a partially known and predictable environment, such as a warehouse or a factory. In these environments, the robot can find unknown fixed and moving obstacles, but such obstacles are mostly predictable, like people, boxes, other robots, etc.

II. Related Work

This section presents some forecast algorithms and mechanisms related to mobile robotics. There are also several studies that propose prediction algorithms for tasks response times and systems performance. Some of these works are also presented in this section.

A. Prediction Mechanisms in Mobile Robotics

Some mechanisms have been proposed to predict the best possible path for the robot with a specific purpose, such as to minimize battery consumption, navigate safely in unknown environments, intercept moving targets, etc.

There are studies that address the problem of navigating a robot in an environment with multiple moving obstacles. These works propose mechanisms that attempt to predict a possible path to avoid moving obstacles, through a heuristic method [1], [2]. The movement of each obstacle is initially predicted assuming they have a constant speed. The algorithm is iterated frequently to accommodate the actual changes in the speed of the obstacles. Based on the movement of the obstacles, the mechanism tries to forecast the robot best navigation speed and then stores the state of the obstacles and the robot speed predicted. If the robot is again faced with a situation previously stored, the corresponding speed is then used. Another mechanism is proposed in [3] and is suitable for environments with static and moving obstacles. The mechanism tries to predict the future movement of the moving obstacles and then plan a safe and efficient path for the robot. To achieve prediction, the mechanism considers the uncertainty in motion planning and uses a probabilistic model of the uncertainty and select the motion which minimizes the expected time of reaching the destination. A method for autonomous mobile robots to avoid moving obstacles by predicting the movement of the obstacles is shown in [4].

An algorithm to intercept moving targets is proposed in [5]. The robot can intercept a target by following many short and straight paths. A point of intersection is initially predicted assuming that the robot and the target move along straight paths. The algorithm then tries to plan a path to the intersection point predicted before. The robot navigates along the planned path, while continuously monitors the target. When the robot detects that the target is in a new position, the mechanism predicts a new intersection point and plans a new navigation path. This process is repeated until the robot intercepts the moving target.

A mechanism to find energy efficient paths for mobile robots is presented in [6]. The mechanism tries to forecast the best paths and finds the recommended speeds for the robot. The relationship between the speed of the motor and its power consumption is modeled through polynomials. The speed of the robot is related to the speed of its wheels through a linear transformation. The algorithm makes a comparison between the energy consumption of different paths at different speeds. Thus, the mechanism can predict the best path that the robot should follow and the recommended speed to minimize energy consumption. A similar mechanism is presented in [7]. An approach to explore unknown environments by a robot in a energy-efficient manner is presented in [8]. The proposed algorithm determines the next target that the robot should reach. The mechanism then finds a path from the current position of the robot to the next target efficiently in terms of energy consumption and avoiding repeated targets. In the end, the mechanism can predict the best combination of paths and the correct order of targets to explore the entire environment, consuming minimal power. In [9] is shown a case study of the major energy consumers in a mobile robot and a method is proposed to estimate the battery life and predict energy consumption moving the robot from an initial position to a final one. In [10] is presented a mechanism to minimize the energy consumption of a mobile robot, simultaneously predicting the best speed to move the robot and the frequency of its board processor. In [11] a mechanism is shown to predict the energy needed for unmanned ground vehicle missions and update the forecast during mission execution through real-time measurements of energy consumption, vehicle speed and other parameters.

B. Prediction Mechanisms for Other Computational Systems

Mechanisms to predict the response time and systems performance are widely studied in computer science. The following are some of these mechanisms.

Three deadline missing prediction mechanisms for real-time systems based on distributed threads are presented in [12]. The first mechanism, is based on the concept of *Milestones* and uses known information of past activations to define Milestones statically in the source node, before the execution of the distributed thread. The second mechanism considers all possible paths that a distributed thread can execute as well as information from previous executions. The prediction is made at runtime, by querying a data structure, which is loaded by the thread as it visits the nodes. The third mechanism relates known information of past activations with information known at runtime, using linear regression. The mechanism also queries a data structure to make the prediction.

A service management system that can predict the response time of transaction-oriented web applications is shown in [13]. The prediction service is initiated by a client which requests to the service manager the response time of a given service. The manager interacts with the system, predicts the response time based on the current system conditions and returns this information to the customer.

A self-prediction mechanism for distributed storage systems is proposed in [14]. The algorithm can predict the performance of a given load if its data are moved from device A to device B. The system generates information used by administrators for the purpose of performance analysis.

In [15] is presented a deadline missing mechanism for embedded systems running several different applications. An application consists of a set of services. Tasks with non-critical deadlines require a service from the application. The algorithm determines the probability of a task meet its deadline, joining all possible states of the system in only two: normal or overloaded. A data structure stores the average response time of each application service. When the task finishes its execution, its response time is compared with the average response time of the corresponding service.

An algorithm to predict the execution time of parallel applications is presented in [16]. The prediction is based on past execution times of similar parallel applications. The algorithm considers applications that do not have temporal limitations.

In the field of distributed real-time systems, functions to predict the response time of periodic processes are shown in [17]. The prediction is made offline and it considers the execution time spent on each activity of the process. The functions used to estimate the response time are based on the period of the processes. In [18] different aproaches are proposed to predict the response time of a real-time distributed application in a node. The proposed methods use *Profiling* to determine the execution time of the application.

III. The prediction mechanism

This paper proposes a deadline missing prediction mechanism for mobile robots performing high-level tasks. These tasks consist in moving the robot from an initial position to another position and back to the original position. Given a particular task and a deadline, the mechanism should be able to predict in a timely manner if the robot is able to complete the task within the given deadline, with a success rate of at least 90%. The mechanism is suitable for any robot with a hybrid paradigm and doesn't depend on the implementation of the path planning and navigation algorithms.

The mechanism predicts if the deadline will be missed when the robot reaches the goal position, before returning to the initial position. This is important because if the mechanism predicts a deadline missing, the task can be canceled or other actions can be executed before the completion of the task.

The proposed mechanism works by collecting information during the execution of tasks and comparing these with other stored information from previous tasks. A history containing information from previous executions is stored and then consulted during the prediction process. The mechanism works in two distinct modes: the learning mode and the prediction mode. The learning mode is used when the history size is not large enough to produce reliable results. In this case, the robot must perform tasks while information is stored in the history until it reaches the size needed for reliable predictions. The prediction mode is the main mode and is used when there are already enough information stored in the history. During navigation, data are collected by the mechanism and compared with the data stored in the history from previous executions.

In the end, the result of the deadline missing prediction algorithm will be based on a formula whose result will indicate whether or not certain task will be completed within a given deadline.

A. The History

As mentioned before, the mechanism uses historical data, represented by a data structure which is consulted during the forecasting process. The history has a defined maximum size and when this size is reached, the oldest information is replaced by newer information. The history stores information such as the initial position of the robot, the goal position, the partial time (time to reach the goal position), the total execution time of the task, the number of known obstacles in the environment and their positions and the average robot speed. Whenever a task is completed, the history is updated with its information if no unknown obstacle was found along the robot's path.

The history is consulted when the robot receives the indication of a new task, before it starts moving. Important data for the prediction algorithm are obtained from the history and then the task is started. The data are used to predict the deadline missing, as soon as the robot reaches the goal position. It is important to note that the data stored in the history at the end of a task, are available for subsequent executions of other tasks. An example of a history is shown in Table I. "Home" is the initial position of the robot, "Goal" is the goal position, "PT" is the partial time in seconds, "TT" is the total execution time of the task in seconds, "NKO" is the number of known obstacles in the environment, "Vel" is the average speed of the robot while performing the task in milimeters per second and "Obstacles" are the positions of the known obstacles in the environment.

TABLE I
An Example of the history

Home	Goal	PT	TT	NKO	Vel	Obstacles
(123,664)	(987,1245)	47.1	105.6	5	256	(873,232)...
(435,344)	(667,-7463)	52.8	120.2	6	340	(987,-456)...
(123,664)	(987,1245)	46.8	106.2	5	253	(873,232)...
(435,344)	(667,-7463)	53.4	121.5	6	335	(987,-456)...
(123,664)	(667,-7463)	56.8	134.2	7	283	(873,232)...
...	...	...	...	...	...	...

B. The Prediction Formula

The deadline missing prediction is calculated as soon as the robot reaches the halfway point (the goal position) through the result of a formula. To this end, some important data are obtained from the history, immediately before the beginning of the task. As mentioned before, the history stores information related to tasks in which the robot did not find any unknown obstacle in its way. A search is performed to return the historical records of executions of tasks with the same initial position, goal position and the arrangement of known obstacles. With these records, the algorithm calculates the average partial and final time of execution, which are the average time the robot takes to reach halfway (the goal position) and the average time it takes to complete the task and return to the original position without finding any unknown

obstacle, respectively. This shows that the history size has a direct influence on the result of the prediction, as the more records are returned, the greater the precision of the calculated average times. The influence of the history size is shown later in this paper.

During the execution of a task, some data are constantly monitored, such as the execution time and the robot velocity. At the prediction time, the mechanism assumes that the fixed unknown obstacles found on the way to the goal position will also be found on the way back to the initial position. The mechanism also assumes that the moving obstacles found along the robot's path to the goal position, will not be found on the way back, but reserves a portion of time, in case any of those obstacles has not completely left the way. The mechanism uses the sensors of the robot (preferably laser, if available) to detect whether unknown obstacles are fixed or not.

Once the robot reaches the goal position, the mechanism uses the partial execution time as well as the estimated time to avoid uknown obstacled and the average times retrieved from history to calculate the deadline prediction as shown in (1). "DL" is the deadline and "CET" is the current execution time. "TAT" and "PAT" are the total and partial average times retrieved from the history, respectively. "TAFO" is the estimated time for the robot to avoid fixed unknown obstacles while returning to the initial position and "TMO" is the amount of time reserved to compensate in case the moving obstacles have not completely left the path of the robot.

$$DL - CET >= TAT - PAT + TAFO + TMO \quad (1)$$

The idea of the formula is to check whether there is enough time for the robot to return to the initial position without missing the given deadline, basing the decision on past executions. The left side of the formula calculates the remaining time of the deadline by subtracting the current execution time from the deadline. The right side of the formula calculates the estimated time for the robot to go back to the initial position. To make this calculation, the mechanism first calculates the average time it takes the robot to return to the initial position without the presence of unknown obstacles along the way (TAT - PAT). The mechanism considers that the time it took the robot to avoid fixed unknown obstacles from the initial position to the goal position will be approximately the same as the time required to avoid these same obstacles on the way back and adds this time to the previous result. This time is calculated by subtracting the average time the robot takes to reach the goal position without finding unkonwn obstacles from the current execution time, as shown in (2).

$$TAFO = CET - PAT \quad (2)$$

Finally, the algorithm adds the time reserved to compensate for the presence of moving obstacles on the way back to the initial position. The time reserved for each moving obstacle may have a small predetermined and fixed value or calculated at runtime. "TMO" is the sum of all those times. If the robot reaches the goal position without encountering any unknown obstacle, the values of TAFO and TMO will be zero and the prediction formula is simplified as shown in (3).

$$DL - CET >= TAT - PAT \quad (3)$$

If the result of the prediction formula is true, it means that the robot will most likely be able to meet the deadline. Otherwise, the deadline will be missed and corrective actions can be executed to avoid or minimize possible losses.

IV. The auxiliary mechanism

As mentioned before, the prediction mechanism supposes that the same fixed unknown obstacles found during the way to the goal position will be found again on the way back to the initial position. As the tasks performed by the robot usually take little time, this scenario is more common to happen. However, in some cases, there may be sudden changes in the environment and the robot can find a different number of obstacles during the way back to the initial position, which may lead to some erroneous predictions. To compensate for possible sudden change in the environment and improve the predictions, we developed an auxiliary mechanism. This mechanism uses a different history which stores some information from past tasks. Every known and mapped environment has a corresponding history which can be consulted to notice changes in the environment and correct predictions.

The history stores a few information at the end of each task in which unknown obstacles were found by the robot. Table II shows the structure of the auxiliary mechanism's history.

TABLE II
An example of the history used by the auxiliary mechanism

Date/Time	Home	Goal	DifObstacles	DifTime
Mon Mar 10 09:04 pm	(425, -8778)	(6255, 766)	+1	+7,2 s
Mon Mar 10 07:23 pm	(-234, 678)	(6255, 766)	-1	-9,3 s
Mon Mar 10 10:04 pm	(425, -8778)	(6255, 766)	0	+0,3 s
Mon Mar 10 10:14 pm	(425, -8778)	(6255, 766)	+1	+8,6 s
Tue Mar 11 09:02 am	(-234, 678)	(-444, 234)	0	-0,4 s
Tue Mar 11 12:15 pm	(425, -8778)	(6255, 766)	+1	+8,4 s
Wed Mar 12 11:04 pm	(425, -8778)	(-444, 234)	+2	+15,3 s
...	...	...	...	...

"Date/Time" represents the date and time the task was started. "Home" is the initial position of the robot and "Goal" is the goal position. "DifObstacles" is the difference between the number of unknown fixed obstacles encountered by the robot on the way to the goal position and back to the initial position. The value is zero if there was no difference, *+k* if the robot found *k* more obstacles then expected while returning to the initial position and *-k* if the robot found *k* less obstacles. "DifTime" is the difference between the estimated time for the robot to return to the initial position from the goal position

(the right side of the prediction formula) and the actual time the robot took to make it. The idea is to store the difference between the estimated time and the actual execution time, caused by the obstacles found while returning to the initial position.

The history of the auxiliary mechanism can be consulted with different criteria. The mechanism selects the history which corresponds to the environment in which the task is being executed and performs a search considering only the records related to similar tasks (same home position and same goal position). The criteria for the search are the following:

- **Latest N performed tasks:** The mechanism searches for the latest N executions of the task. The idea is that the environment is probably similar to what it was in the latest task executions. If N=10 for example, the mechanism verifies the lastest 10 executions of the task. If N = 1, the mechanism only selects the last execution and relies on it.
- **All the performed tasks:** All similar tasks to the current one are retrieved from the history.
- **Tasks performed on a time interval:** The mechanism selects specific task executions considering the date/time they were started. In a factory or warehouse, there are peak times when the environment changes frequently and times of little or no activity. This information is important because it can directly influence the results of the deadline missing prediction. So the mechanism can consider only executions of a task in a given time range. The filter can be further restricted and only consider executions of tasks in a time range on a specific week day or a specific month, if the month is relevant.

A query to the history returns a set of records with relevant information from similar tasks. These records are used to update the value of TAFO (estimated time for the robot to avoid fixed unknown obstacles) in (1). The mechanism verifies what happened in the majority of the tasks and updates the value of TAFO according to the result. If in the majority of the executions, the robot encountered the same obstacles while going to the goal position and returning to the initial position, then probably in the current task, the same will happen and the prediction mechanism is likely to succeed. Otherwise, the TAFO value is updated with the average ”DifTime”, i.e, the mechanism adds to TAFO the average time that differs from the estimated time calculated by the prediction formula. The average time calculated can have a negative or positive value, depending on whether the robot reached the initial position before or after the estimated time, respectively, in the majority of the executions. In the example shown in Table II, if in the current task the home position has coordinates (425,-8778) and the goal position has coordinates (6255,766) and if the search criterion is the first one with N=4, we find that in 3 of the latest 4 executions, the robot found an extra obstacle that was not expected, while returning to the initial position. The mechanism considers that the robot will probably find an extra obstacle again and adds 8.06 seconds to TAFO [(7.2s + 8.6s + 8.4s) / 3].

The auxiliary mechanism can be enabled and disabled as needed to correct erroneous predictions due to frequent changes in the environment.

V. Simulations

To validate the mechanism, several simulations were performed in different conditions, with different arrangement of obstacles and different history sizes. First we made several simulations in a static and simulated environment using the robot simulator. After confirming that the mechanism works correctly in simulated environments, we used a real environment and the actual robot for testing.

The Pioneer 3-DX is a fast, smart and versatile mobile robot, widely used for research and experiments in the field of mobile robotics. It has become an increasingly reference platform to implement and compare different algorithms. The P3-DX has a rigid aluminum body and offers an embedded computer (with Linux Operational System) that has many components that a user can program according to his application needs. Some of these components are resources for vision processing, communication via Ethernet (wired and wireless), sixteen sonars to detect objects located between 15 cm and 7 meters away, four big wheels coupled with powerful engines capable of a maximum speed of 1.6 meters/second and carry a weight of up to 23 kg, bumpers with touch sensors and 3 batteries that guarantee an autonomy of 8-10 hours.

There are a variety of accessories and sensors that can be coupled to increase the versatility of the robot. One of the most important sensors for precise localization of the robot and path planning is the laser sensor.

There are different software platforms used for the simulations. These platforms are used to control and simulate the Pioneer 3-DX robot and attached sensors. The main platform used to control and program the robot is ARIA. ARIA provides several classes and functions for various operations and provides simple commands for controlling various complex maneuvers of the robot. Another software used is ARNL, which is an extension of ARIA for localization, navigation, path planning and communication across the network.

MobileSim, the Pioneer robot simulator is also used to test the proposed mechanism before testing on the actual robot. It can simulate the behavior of the robot and its sensors, such as the laser. The Mapper3 software is used to generate the map of the environment. Finally, MobileEyes platform provides an easy to use graphical interface for path planning and localization tasks. Fig. 1 shows a simulation running on MobileSim.

A. Simulated Environment

In this case, the simulated environment is static, i.e, contains only known and fixed obstacles. We used three different configurations of the environment, with five obstacles in different positions.

In each configuration of the environment, we performed 100 different simulations, varying the history size. We used histories with 50, 200, 600 and 1000 records. For every history size we executed 100 simulations, totaling 400 simulations

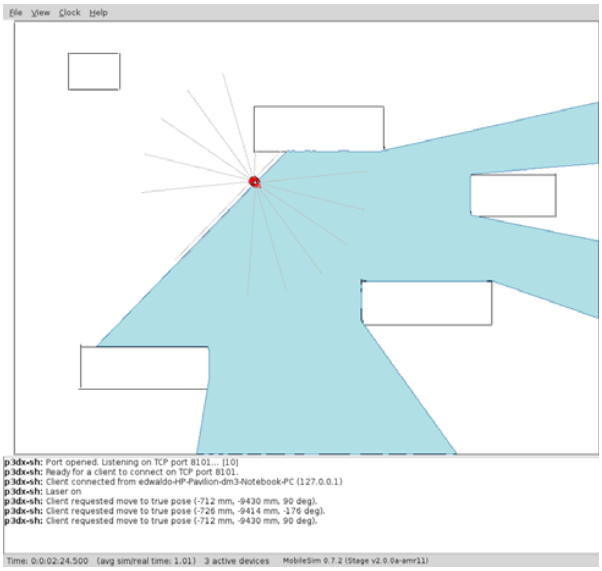

Fig. 1 Simulation running on MobileSim

for each configuration of the environment and 1200 total simulations. Besides showing that the mechanism works, we wanted to show that the history size has influence on the quality of the prediction. Fig. 2, Fig. 3 and Fig. 4 show the three different configurations used. First, we executed the mechanism in learning mode repeatedly until the history reached the desired size. Then we enabled the prediction mode and performed simulations.

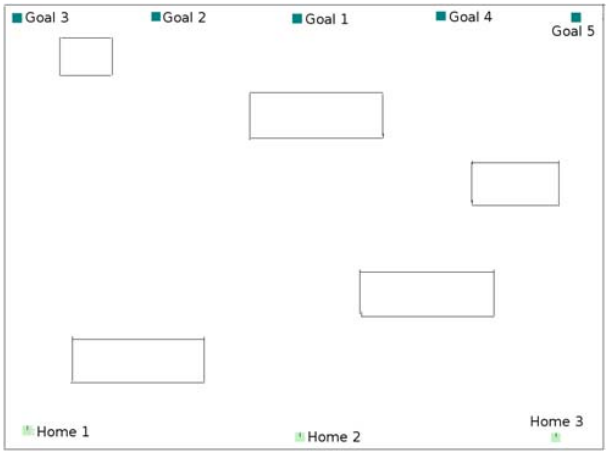

Fig. 2 First Configuration of the environment

The simulations were performed using MobileSim. The deadline used in the simulations is the final average execution time, calculated from the history immediately before each task execution.

1) Results: The simulations results illustrate the main features of the proposed mechanism. They showed that the mechanism works, is reliable and ensures a correct prediction rate greater than 90%. Table III shows the minimum, average and maximum execution times (in seconds) in simulations with

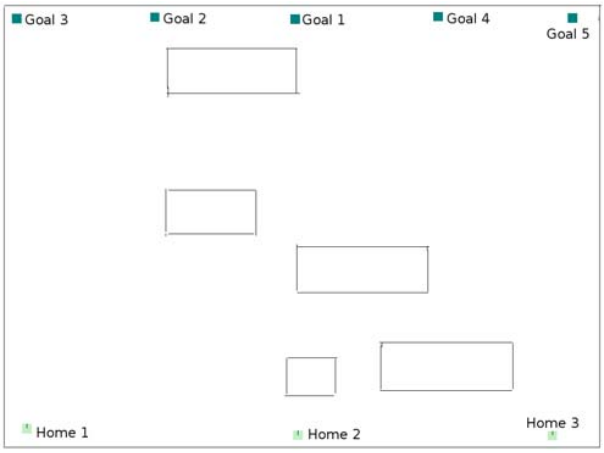

Fig. 3 Second Configuration of the environment

1000 records in the history.

TABLE III
SIMULATION RESULTS. HISTORY SIZE = 1000

		Minimum	Average	Maximum
C1	Partial Execution Time	40.97	42.56	45.12
	Total Execution Time	99.95	102.24	104.73
C2	Partial Execution Time	41.22	44.23	45.78
	Total Execution Time	102.23	104.32	106.23
C3	Partial Execution Time	43.83	46.33	48.56
	Total Execution Time	105.34	108.77	110.97

As expected, the quality of the deadline missing predictions depends on the size of the history. With only 50 entries in the history, we obtained a correct prediction rate ranging from 52% to 62%. With 200 and 600 entries, the rate ranged from 72% to 77% and from 84% to 87%, repectively. Finally, with 1000 records in the history, the correct prediction rate exceeded 90%. Fig. 5 shows the correct prediction rate resulting from all the simulations in each configuration of the environment. Fig. 6 shows the influence of the history size on

Fig. 4 Third Configuration of the environment

the rate. Since we could exceed 90%, it was not necessary to further increase the size of the history, but this can be done to improve the correct prediction rate, if necessary. We believe that 1000 is a number that brings a good balance between performance and accuracy.

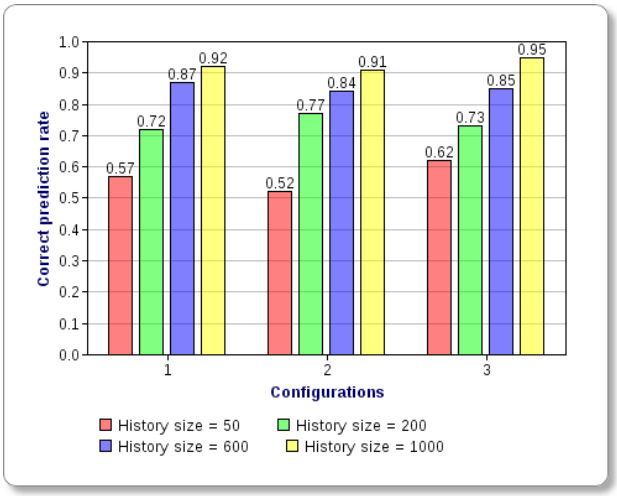

Fig. 5 The correct prediction rate in sumulations

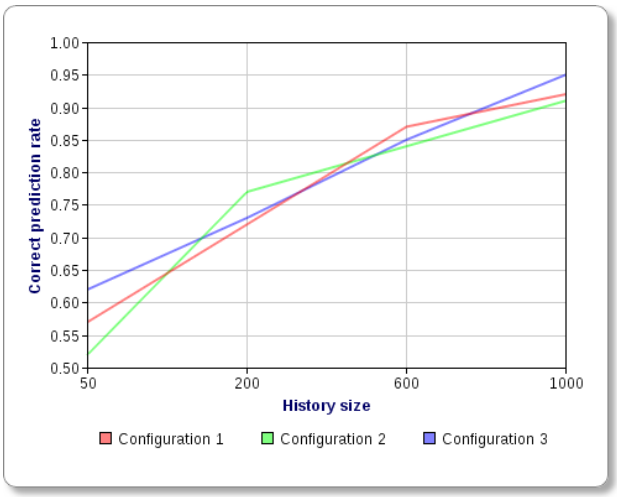

Fig. 6 The influence of history on the correct prediction rate

B. Real Environment

After the tests in the simulated environments we could verify that the mechanism works and is effective. So, we decided to make simulations in the real world, using the real robot. The first step before executing the simulations was to create the map of the desired environment. The environment chosen is the corridor of one of the floors of a building in our university. To create the map, the Pioneer 3-DX robot was teleoperated with a joystick around the corridor while the laser scanned the environment. This process resulted in the creation of a log file, which was opened in the Mapper3 to generate the real map of the environment. Fig. 7 shows the resulting map.

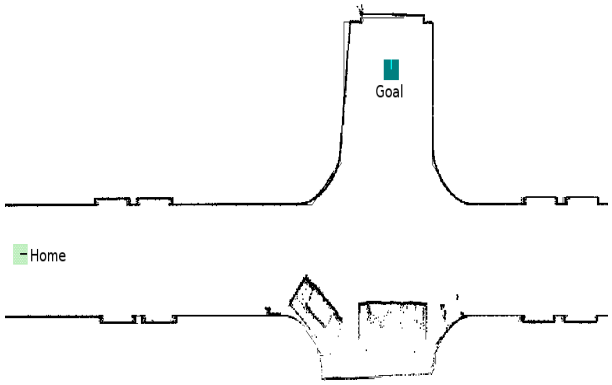

Fig. 7 The map of the environment

After creating the map, we created a history with size=100. Our environment contains only one home position and one goal position so a history with size=100 is enough for a high rate of correct predictions.

Then we executed the mechanism in learning mode 100 times to fully populate the history. Once the history was full, we decided that for each simulation the deadline would be a random value varying in the range [TAT-30s, TAT+30s]. As mentioned previously, TAT is the total average time retrieved from the history, i.e, the average time that the robot usualy takes to complete the task without finding any unkown obstacle. Then we disabled the auxiliary mechanism and we executed the mechanism in the prediction mode 100 more times. During these simulations, we placed unknown obstacles like chairs along the robot's path and we walked in front of the robot a few times to simulate moving obstacles.

We checked the results and we made the mechanism fail a few times deliberately by putting extra unknown obstacles in the robot's path to the initial position. Finally, we enabled the auxiliary mechanism with the first search criterion and N=5 and we checked the prediction results after executing the prediction mechanism a few more times. Fig. 8 and Fig. 9 show the robot performing tasks and avoiding unknown obstacles.

1) Results: The simulation results showed that the mechanism also works in real environments with unknown obsctacles, is reliable and ensures a correct prediction rate greater than 90%.

After the 100 executions of the mechanism in prediction mode, we managed to get a correct prediction rate exceeding 90%, as intended. The results are shown in Fig. 10.

By placing various unexpected obstacles in the robot's way back to the initial position, we managed to force a few wrong predictions, as intended.

After we enabled the auxiliary mechanism with N=5 and run the forecast mechanism a few more times, the results were as expected. The mechanism realized that the robot was finding two more obstacles than expected. The mechanism started to adjust the value of TAFO with the data obtained by consulting

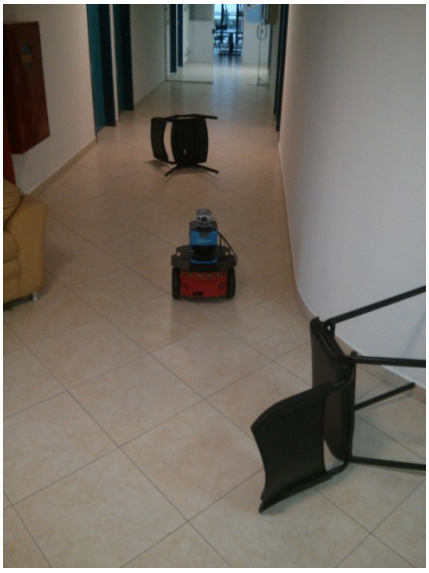

Fig. 8 Robot Pionner 3-dx performing a task

Fig. 9 Robot Pionner 3-dx avoiding an unknown obstacle

the history and the main mechanism started to produce correct predictions again.

VI. Conclusions and Future Work

Nowadays, mobile robotics is gaining an increasingly important role in modern society. Mobile robots are increasingly useful in many areas of our lives such as entertainment, medicine, industry, military, research, etc. Human labor is being replaced by intelligent and autonomous robots that can easily perform dangerous and stressful tasks for humans. Many robots are designed to perform useful services to humans and they assist in several tasks such as rescue, domestic assistance (vacuum cleaners, lawn mowers, etc.),

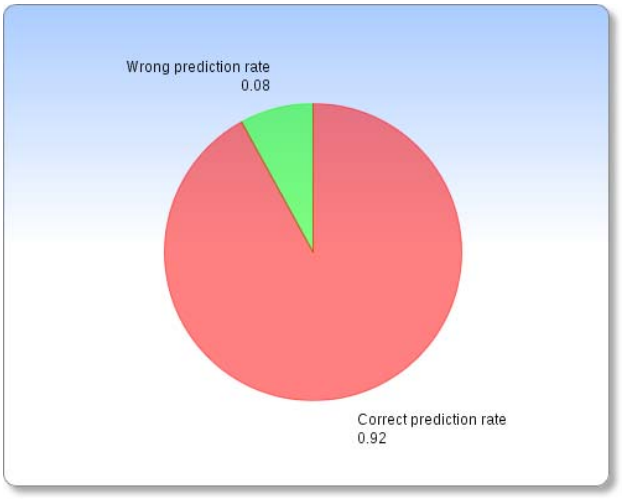

Fig. 10 The correct prediction rate in a real environment

entertainment, assistance to persons with disabilities, among others. The practical application of mobile robots in different activities in our society, has shown how promising is the future of mobile robotics.

We live in a tech society, and it is estimated that in the future, human being will be responsible only for tasks that require essentially human functions such as thinking, imagining, reasoning, creating, designing, etc. Robots will not be able to perform these tasks autonomously, for a while.

Real-time system's temporal constraints can be applied to high-level tasks of mobile robots in the situations where tight deadlines are required for proper system operation. Missing a deadline can result in material and financial losses. The environment in which a robot moves around and performs tasks can be dynamic and not always predictable. This means that it is not always possible to predict statically whether a robot can perform a task within a given deadline. In this sense, it is necessary to develop mechanisms to predict deadline missing at runtime.

This paper presented a simple and reliable deadline missing prediction mechanism for mobile robots and demonstrated its effectiveness when used both in simulated environments and real environments.

While performing tasks, the robot can find unknown obstacles along the path to the target position. On the way back to the initial position, if the robot find those same fixed obstacles, the mechanism hardly makes a wrong prediction. An auxiliary mechanism was developed to compensate for possible failures resulting from the sudden changes in the environment. So far, the user is responsible for enabling the auxiliary mechanism when needed and for selecting the criterion that should be used in the search. The next step in this work is to improve the auxiliary mechanism and automate its use. The prediction mechanism itself can enable the auxiliary mechanism when needed (e.g, when there are several wrong predictions) and select the best criteria as well as disable it if not needed. Unexpected moving obstacles encountered on the way back to the initial position tend not to have much influence

on the result of the prediction, because in general, the time that the robot takes to avoid these obstacles is considerably smaller than in the case of fixed obstacles.

Some information stored in the history such as the velocity were not used in this work, but may be used later, as well as any other information that might be relevant to improve the mechanism. The proposed prediction mechanism uses the localization, navigation and path planning algorithms developed by the robot manufacturer. These algorithms are proprietary, but are fast and efficient. The mechanism can be adapted to work with any robot or development environment, but its efficiency depends on the sensors used by the robot to detect obstacles and the quality of the localization and path planning algorithms. The laser sensor offers far greater precision in the localization and detection of obstacles than sonar.

REFERENCES

[1] T. Tsubouchi and S. Arimoto, "Behavior of a mobile robot navigated by an ldquo;iterated forecast and planning rdquo; scheme in the presence of multiple moving obstacles," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, 1994, pp. 2470–2475 vol.3.

[2] T. Tsubouchi, A. Hirose, and S. Arimoto, "A navigation scheme with learning for a mobile robot among multiple moving obstacles," in *Intelligent Robots and Systems '93, IROS '93. Proceedings of the 1993 IEEE/RSJ International Conference on*, vol. 3, 1993, pp. 2234–2240 vol.3.

[3] J. Miura, H. Uozumi, and Y. Shirai, "Mobile robot motion planning considering the motion uncertainty of moving obstacles," in *Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics*, 1999, pp. 692–697.

[4] C. Shi, Y. Wang, and J. Yang, "A local obstacle avoidance method for mobile robots in partially known environment," *Robot. Auton. Syst.*, vol. 58, no. 5, pp. 425–434, May 2010.

[5] Q. Zhu, J. Hu, and L. Henschen, "A new moving target interception algorithm for mobile robots based on sub-goal forecasting and an improved scout ant algorithm," *Applied Soft Computing*, vol. 13, no. 1, pp. 539 – 549, 2013.

[6] Y. Mei, Y.-H. Lu, Y. Hu, and C. S. G. Lee, "Energy-efficient motion planning for mobile robots," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 5, 2004, pp. 4344–4349 Vol.5.

[7] M. Jia, G. Zhou, and Z. Chen, "An efficient strategy integrating grid and topological information for robot exploration," in *Robotics, Automation and Mechatronics, 2004 IEEE Conference on*, vol. 2, 2004, pp. 667–672 vol.2.

[8] Y. Mei, Y.-H. Lu, C. S. G. Lee, and Y. Hu, "Energy-efficient mobile robot exploration," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006, pp. 505–511.

[9] Y. Mei, Y.-H. Lu, Y. Hu, and C. S. G. Lee, "A case study of mobile robot's energy consumption and conservation techniques," in *Advanced Robotics, 2005. ICAR '05. Proceedings., 12th International Conference on*, 2005, pp. 492–497.

[10] W. Zhang, Y.-H. Lu, and J. Hu, "Optimal solutions to a class of power management problems in mobile robots," *Automatica*, vol. 45, no. 4, pp. 989 – 996, 2009.

[11] A. Sadrpour, J. Jin, and A. Ulsoy, "Mission energy prediction for unmanned ground vehicles," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 2229–2234.

[12] P. Plentz, "Mecanismos de previsão de perda de deadline para sistemas baseados em threads distribuídas tempo real," Doutorado, UFSC, Florianopólis - SC, 2008.

[13] S. Kirtane and J. Martin, "Application performance prediction in autonomic systems," in *Proceedings of the 44th annual Southeast regional conference*, ser. ACM-SE 44. New York, NY, USA: ACM, 2006, pp. 566–572.

[14] E. Thereska, M. Abd-El-Malek, J. Wylie, D. Narayanan, and G. Ganger, "Informed data distribution selection in a self-predicting storage system," in *Autonomic Computing, 2006. ICAC '06. IEEE International Conference on*, 2006, pp. 187–198.

[15] C. Tatibana, C. Montez, and R. Oliveira, "Soft real-time task response time prediction in dynamic embedded systems," in *Software Technologies for Embedded and Ubiquitous Systems*, ser. Lecture Notes in Computer Science, R. Obermaisser, Y. Nah, P. Puschner, and F. Rammig, Eds. Springer Berlin Heidelberg, 2007, vol. 4761, pp. 273–282.

[16] W. Smith, I. Foster, and V. Taylor, "Predicting application run times with historical information," *J. Parallel Distrib. Comput.*, vol. 64, no. 9, pp. 1007–1016, Sept. 2004.

[17] L. Welch, A. Stoyenko, and T. Marlowe, "Response time prediction for distributed processes specified in cart-spec," *Control Engineering Practice*, vol. 3, no. 5, pp. 651 – 664, 1995.

[18] E.-N. Huh and L. R. Welch, "Adaptive resource management for dynamic distributed real-time applications," *J. Supercomput.*, vol. 38, no. 2, pp. 127–142, Nov. 2006.