In appear in: *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium*, Toronto, CA, May 2004.

A Real-Time Model for the Robotic Highway Safety Marker System*

Jiazheng Shi Steve Goddard Anagh Lal
Computer Science & Engineering
University of Nebraska—Lincoln
Lincoln, NE 68588-0115
{*jshi,goddard,alal*}*@cse.unl.edu*

Shane Farritor
Mechanical Engineering
University of Nebraska - Lincoln
Lincoln, NE 68588-0656
{*sfarritor*}*@unl.edu*

Abstract

This paper presents the design and implementation of a real-time model for the global control of robotic highway safety markers. Problems addressed in the system are: 1) poor scalability and predictability as the number of markers increases, 2) jerky movement of markers, and 3) false-hits caused by environment objects.

We extensively analyze the system and offer two solutions: a basic solution and an enhanced solution. They are built respectively upon two task models: the periodic task model and the variable rate execution task model. The former is characterized by four static parameters: phase, period, worst case execution time and relative deadline. The latter has similar parameters. Its parameter values, however, are allowed to change at arbitrary times. We then examine two typical real-time scheduling approaches: rate monotonic (RM) priority driven and earliest deadline first (EDF). Analysis of their sufficient conditions shows that our system is feasibly schedulable under either RM or EDF. This conclusion justifies that the path for each safety marker can be guaranteed to be smooth under the designed real-time system. For the scalability issue, we present a sufficient condition for the upper bound on the number of barrel robots that can be reliably controlled.

One key technique integrated into our real-time system is the Hough transform. We refine its traditional implementation so that, for the task of detecting safety markers, the time complexity decreases and the reliability increases with only slight additional storage for search windows. The basic idea behind our improvements is to limit the search window for safety markers.

*Supported, in part, by grants from the National Science Foundation (grant CCR-0208619) and the National Academy of Sciences Transportation Research Board-NCHRP IDEA Program (Project #90).

1 Introduction

Robotic highway construction and maintenance has been a research topic since the 1990's. Throughout this period, people have presented several practical solutions for their specific applications [23, 29, 30, 27, 12]. Our application is the Robotic Highway Safety Marker system [7, 25]. The basic idea is to develop an autonomous, robotic, real-time system that automates the placement of highway safety markers in hazardous areas, thereby eliminating risk to human workers. Our solution is especially useful in those environments where a limited number of barrels need deploying and they are moved frequently, e.g., patch a pothole every 100 meters [25]. Figure 1 shows the working field for a system test. The stripped plastic barrels are safety markers.

Low system cost, high reliability, and timely deployment are three major considerations to the Robotic Highway Safety Marker system. Their performance depends on both software and hardware design. Thus, technologies exploited in the whole system cover several disciplines, including mechanical engineering, electrical engineering, software engineering and real-time systems. The scope of this paper, however, is confined to the software design of the real-time system incorporated in a foreman (also called a lead robot in this paper), whose detailed functions are presented in the following subsection. This paper presents several practical methods of real-time systems design to lower the cost, increase the reliability and ensure timely deployment of markers.

In the following two subsections, the system design satisfying the above three requirements is presented first. We then identify exact problems challenging our software design of the foreman, from a real-time perspective.

1.1 System Design

This subsection gives a brief description of the system design. A more detailed description can be found in [25]. The system design is subject to three major considerations:

In appear in: *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium*, Toronto, CA, May 2004.

(A) Working environment (B) Prototype foreman

Figure 1. A working environment (left) and the foreman (right).

low system cost, high reliability, and timely deployment. The system cost limits how successfully our application can be widely applied in practice. Particularly, cost per barrel should be sufficiently low because 1) a working field usually needs to deploy many barrel robots and 2) barrels are prone to being destroyed. The reliability relates to how often the safety barrels can be deployed correctly within a certain time interval and certain working environments. Timely deployment requires that barrel robots be deployed in a desired configuration within as short a time as possible. Appropriate software and hardware design may guarantee the system reliability and timely deployment.

To satisfy the above three basic requirements, a master/slave hierarchical control system is introduced in our system design. The system consists of two types of components: foremen and barrel robots. A foreman (or lead or guiding robot analogous to a master), guides and plans paths for more than one barrel robot, which is analogous to a slave. Each barrel robot is contained inside a barrel shaped safety marker (hence the name barrel robot). The foreman computes a path for each barrel robot under its control, sending way-points at regular intervals to communicate the desired configuration. The barrel robots receive these way-points and compute their respective local paths from the current way-point to the next way-point. Thus, system control can be broken down into two parts: global control which lies in the foremen and local control which is distributed into barrel robots. Actual applications are usually made up of several groups of robots. Within each group, one foreman manages many barrel robots.

This centralized sensing and control can reduce the cost of each barrel robot. In one aspect, only low computational capability and a small amount of memory are required for each individual barrel robot. This is because tasks for local planing and hazard avoidance sensing are not resource demanding. In another aspect, barrel robots are free from expensive transceivers. The communication rate between the foreman and the barrel robots is very low, with a default bandwidth of 19,200bps. A slight cost increase results for the foreman, which requires a sophisticated mobile robot. However, one foreman can control many barrel robots. Analysis in later sections shows a conservative upper bound on the number of barrel robots that one foreman can manage.

Reliability is determined by hardware and software design. The hardware consideration for the barrel robots involves mainly mechanical engineering: 1) stability in high winds ($60mph$), which is possible when a car in the highway passes by the barrel robots, 2) appropriate weight ($< 12kg$), which should ensure not only the stability but also easy movement by workers, 3) ability to climb slopes ($< 7\%$ $grade$), 4) appropriate travel speed as fast as $8km/hr$, and 5) traverse small ($8cm$) obstructions. These hardware requirements for barrels can be satisfied by deliberate design in terms of mechanical engineering. In this paper, however, we focus only on the software reliability and the scope of which is confined to the temporal reliability of the real time tasks in the foremen. For space consideration, we only examine solutions in one foreman.

It is helpful to offer a brief description on the global planning and control implemented in the foreman. The task of global planning and control performs functions to move a group of barrel robots under the management of the foreman from an initial configuration to a desired configuration with two requirements: 1) minimal motion time and 2) easy implementation in the foreman. The major challenge involved in the motion planning is to avoid potential collisions among barrel robots in the working field.

Within the past twenty years, people have presented many approaches to motion planning for mobile robots [3, 11, 2,

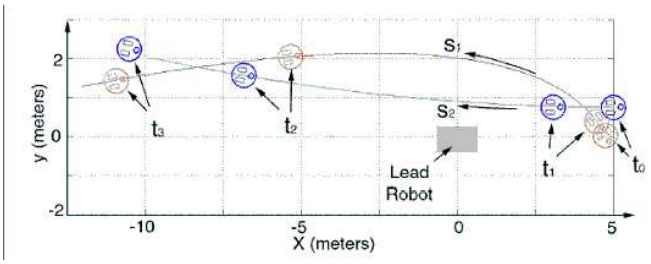


Figure 2. Paths that two barrel robots follow.

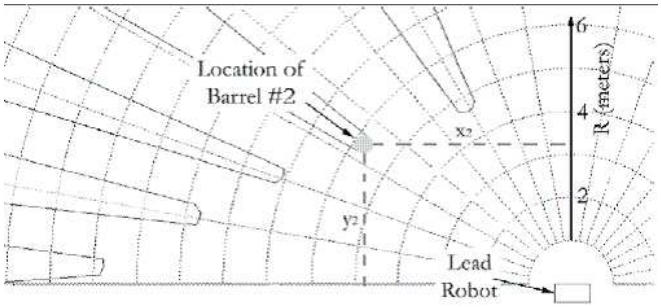


Figure 3. An example of scanning for barrel robots by the foreman (lead robot).

6]. The global planning approach used in our application stems from cooperative industrial manipulators [21]. The difference is that our design incorporates behavior control where high priority is given to the barrel robots that must travel the greatest distance [25].

Figure 2 illustrates the paths that two barrel robots are designed to follow from one location to their respective destinations. Instead of a straight line, the path of each barrel robot consists of piecewise parabolic polynomials, each of which is characterized with three boundary conditions: initial position, initial orientation, and final position. This mathematical model can be justified by the following considerations: 1) to minimize the motion time, each barrel robot should try to keep velocity as high as possible; 2) the avoidance of barrel collision requires that barrel robots change their moving velocity including both speed and direction; 3) a parabolic polynomial, which is computationally effective, can be determined exactly by three boundary conditions. In [25], Shen proves that this design can ensure the minimal time to complete the reconfiguration of the group of barrel robots without barrel collisions.

1.2 Model Problems

Mechanical limitations can cause barrel robots to deviate from pre-computed paths. To correct this, the foreman monitors the movements of the barrel robots, using a laser to scan the working field every 50 milliseconds (ms). Figure 3 illustrates a snapshot of this type of scanning. The foreman then detects each robot and determines its actual position. The way-points sent by the foreman compensate for the difference between the barrel robot's current and expected positions, thereby bringing the barrel robot back on track. The foreman continues scanning, detecting, tracking the barrel robots and sending way-points to them while their desired configurations are achieved.

The processor-intensive detection algorithm described above may keep the foreman from being able to detect and track robots in a timely manner, which results in a stop-and-go motion of the barrel robots. This problem can be solved temporarily by sending intermediate way-points; however, this jerky movement problem is likely to resurface when there are more barrel robots to detect, in which case more intensive computation in the foreman is required. If the way-point computation is made predictable, however, we can guarantee smooth motion of the barrel robots. This is because when receiving regular way-points the barrel robots do not stop until a new path is computed. Instead, they continue in motion, as a new path is available when each barrel robot reaches a destination way-point. In this paper, we discuss how we can guarantee completion of jobs critical to smooth movement using a real-time system model.

The detection algorithm used is a variant of the Hough Transform [24]. The algorithm is efficient but is prone to false-hits, i.e., detecting environment objects as barrel robots. For example, the algorithm sometimes detects the human leg as a barrel robot. Therefore there is a need to make this algorithm more robust and efficient. The detection algorithm is a bottleneck of the deployed system, and an improved detection algorithm is presented here.

The Robotic Highway Safety Marker system is a real-time system because temporal correctness is important in the system for the detecting, tracking and planning phases. The laser scanning is performed periodically. The detection of barrel robots is to be completed within the interval between two scans so that robot positions can be determined at every scan. The foreman plans paths for all the robots. Way-points are to be sent at regular intervals so that the barrel robots do not deviate to an undesirable position. By undesirable position we mean, when a barrel robot goes beyond the communication range of the foreman or, enters a collision path with other barrel robots or environment objects. This is another example where temporal correctness of the system is important. Further, by making the system more predictable through the use of real-time scheduling algorithms, we can improve the quality of the system.

With the appropriate real time model, our three major design considerations, low cost, high reliability, and timely deployment, are achieved in part by solving the following problems:

1. refine the basic Hough-transform based barrel robot detection algorithm, so that detecting time decreases while the reliability increases;

2. build a predictable and robust task model and appropriate scheduling approaches for system tasks;
3. ensure a smooth movement for each barrel robot with minimal deployment time;
4. improve the scalability of the system, which can support various number of barrel robots managed by one foreman. Thus, cost of the system is reduced.

The rest of this paper is organized as follows. Section 2 describes a refined Hough transform, which is the underlying technique adopted to detect barrel robots. Section 3 presents a basic solution for the system. This solution consists of a static task model and a typical scheduling approach. In Section 4, we discuss a more sophisticated solution, which models the tasks in the foreman as a dynamic task set controlled by the refined Hough transform. Section 5 gives conclusions and future work.

2 Hough Transform

A key technique integrated into our system is the Hough transform. Thus, it is helpful to present it first. The Hough transform technique is used widely in computer vision [24]. It performs functions to isolate objects with particular shapes from images. The basic idea behind the Hough transform is to detect shapes by clustering parameter values in a multidimensional parameter space. The classical Hough transform requires that shapes, such as lines, circles, and ellipses be parameterized explicitly. Generalized Hough transform can detect objects that are hard to describe with explicit analysis, though it is computationally demanding [13]. Our application utilizes the classical Hough transform to detect barrel robots which are characterized with circles in images generated by the laser scanner.

The complexity of the Hough transform is governed by two major factors: the shape of objects and the image size. The former factor measures how many parameters (also called dimensions in the computer vision literature) are required to describe the objects. The latter factor measures how many instances for each parameter are generated for clustering. Either high parameter dimension or large image size may increase both time and space complexity of the Hough transform.

In our application, circle objects are described by only three parameters: the 2D location (x, y) and radius. The image size of the basic implementation of the Hough transform is equal to the number of sampling points. Sampling points are the output of the laser scanner which changes its scan angle from 0^o to 180^o by a specified step, e.g. 0.5^o in the default scanner setting. This scan covers a half circle region with a radius be 32.767 meters. For computational convenience, sampling points are saved as (x, y) values in the 2D Cartesian coordinate system, which is a transformation of the original 2D polar coordinate system with the laser scanner being the common origin.

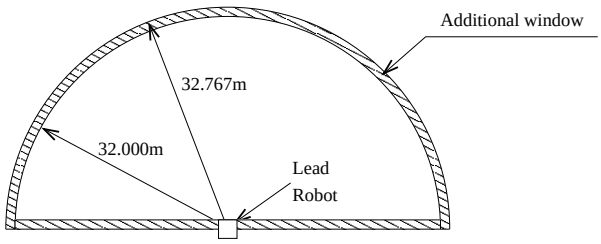


Figure 5. The additional window for the enhanced solution.

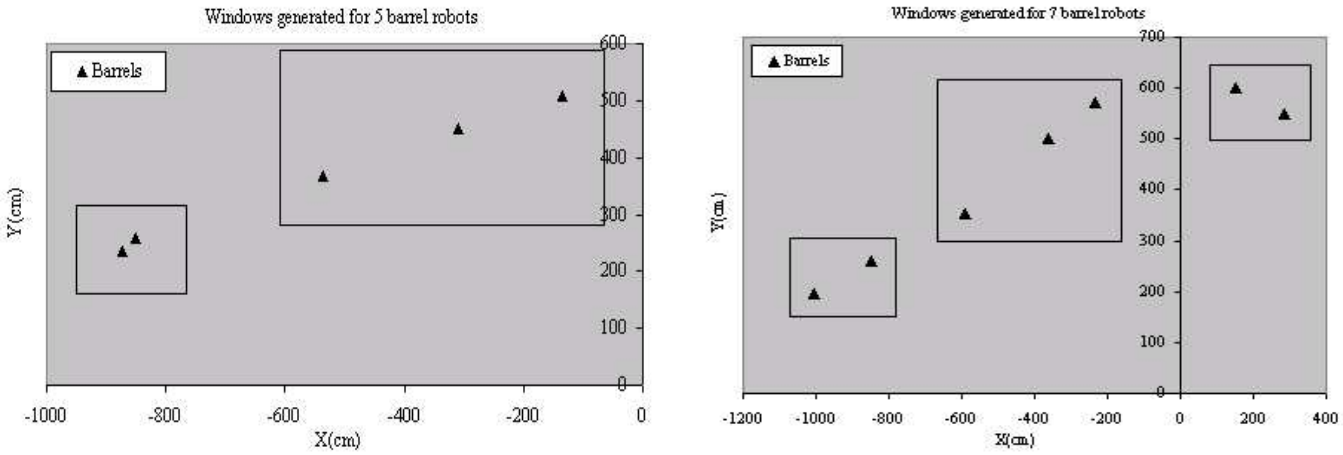
We cannot characterize the barrel shape with less parameters than three to decrease the computational complexity of the Hough transform algorithm. However, we can lower the time complexity by shrinking the search windows because barrel robots are usually clustered in a small region and thus most of the scannable region remains blank.

The basic idea is similar to the Kalman track [26]. We predict search windows where barrel robots are likely to be, based on their previous coordinates. In our refined implementation of the Hough transform, each cluster of barrel robots is grouped within one common window. Depending on the distribution of barrel robots, search windows are updated dynamically. That is, two windows may be merged if barrel robots within these two windows are sufficiently close, and one window may also be split if the distribution of barrel robots within this window is very sparse.

A window resizing task updates windows after every scan. Windows are created by reading sampling points from left to right and from bottom to up in the 2D Cartesian grid. The initial window, which contains only one barrel robot, is a square with 2 meter sides. We predict that, between two successive scans, the displacement of one barrel along the east, south, west, or north is less than 0.75 meters. Whenever the distance between the geometric center of a window and an outside barrel robot is less than 3.5 meters, the window is extended to cover this outside barrel robot. If all outside barrel robots are far from the existing window, a new window is initialized. Figure 4 illustrates two snapshots of windows for 5 and 7 barrel robots respectively.

In our enhanced solution, barrel robots are allowed to leave and enter the scannable region of a foreman during runtime. Thus, we design an additional window which covers exactly the half circle edge of the scannable region. Rather than the rectangle windows, this window is a region as shown in Figure 5.

By introducing this new method for search windows, the time complexity of the Hough Transform can be reduced because most sampling points, which are not involved in barrel robot locations, can be skipped by range checking with the predicted search windows. Table 1 presents results of the



(A) 5 robots in 2 windows (B) 7 robots in 3 windows

Figure 4. A snapshot of refined windows for the Hough transform with 5 and 7 barrel robots.

# of barrels	# of skipped sampling points	execution time(ms)
3	0	2.627
	326	2.588
5	0	2.632
	314	2.599
6	0	2.65
	295	2.641

Table 1. Comparison of the execution time of the Hough transform.

execution time of the Hough transform, using 1,000 independent tests. Results show that:

- the execution time is inversely proportional to the number of skipped sampling points;
- the improvement is not strictly related to the the number of barrel robots.

The first result shows that, in terms of time complexity, the refined Hough transform outperforms its traditional implementation, where no data are skipped. This improvement increases as the distribution of barrel robots becomes more compact. The second result can be explained by the different distributions of barrel robots in the working field. A sparse distribution may increase the number of search windows and thus, decrease the number of skipped data.

Under this refined Hough transform, the number of search windows, N_w, is subject to the number of barrel robots, N_r, within the scanner view. The worst case ($N_w = N_r$) happens when the distribution of barrel robots is very sparse. Thus, we claim that the space complexity of the refined Hough transform is $\Theta(N_r)$, which requires only slight additional storage to the traditional implementation.

The refined Hough transform also outperforms its traditional implementation in detecting reliability. Predicted search windows can be used to reduce the false-hits by adding constraint checks to confirm the detection is of a barrel and nothing else. First, only objects within search windows are detected, so that we can avoid false-hits caused by objects outside search windows. Second, within search windows, only those objects with sufficient dimensions are confirmed to be barrel robots. In our implementation, only N_r objects are stored. They have relatively large dimensions calculated by the Hough transform.

3 Basic Solution by Applying Real-Time Technologies

This section describes a static task model under rate monotonic (RM) scheduling. We focus on tasks only in the foreman. The task model is static because parameters characterizing a task are all fixed before scheduling. The model problem can thus be simplified for analysis convenience. In Section 4, we present a dynamic task model, named variable rate execution (VRE), which allows tasks to enter and leave the system at arbitrary times and supports variable worst case execution time (WCET) and periods [9].

3.1 Static Task Modeling

For task modeling, two important issues are: 1) how to correctly identify tasks that the system requires and 2) how to reasonably estimate parameter values for each task. Regarding estimation, the hardest is the task execution time which relies on both hardware and software components installed in the foreman machine.

Tasks can be described by several parameters. Among them, the most crucial parameters are the task's first release time or phase ϕ, release interval or period p, worst case execution time (WCET) e (including context-switch time,) and

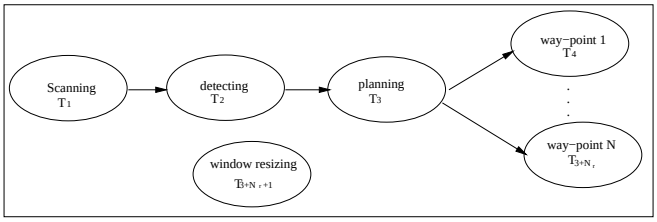


Figure 6. Scheduled tasks existing in the foreman.

relative deadline d. In terms of this 4-tuple (ϕ, p, e, d), tasks are canonically categorized into three reference models: periodic task model, sporadic task model, and aperiodic model. The periodic task model is characterized by an exact release interval between two consecutive jobs. Jobs of a sporadic task are released at arbitrary times and the p only denotes the minimum release interval. Aperiodic tasks have no hard deadline specified for each job, thus people also refer them to as non-real-time tasks.

As introduced in Section 1, the foreman of the robotic highway safety marker system performs the following functions:

1. periodically receive the position data of barrel robots from the laser scanner. This task is named scanning in this paper;
2. periodically detect the positions of all barrel robots. This task is named detecting;
3. periodically plan way-points for all barrel robots. This task is named planning;
4. periodically communicate with each barrel robot to update way-points. Let N_r be the number of the barrel robots surrounding the foreman. There exist N_r tasks, named way-point 1,..., way-point N_r;
5. resize the search window of the Hough-transform. This task is used for performance improvement of the Hough transform. It is named resizing;
6. provide an interface occasionally for human workers. These tasks are aperiodic. Task modeling and scheduling approaches presented in this paper do not account for them.

The foreman system is mixed with real-time periodic tasks: task T_1 through task T_{3+N_r} and a non hard real-time task T_{3+N_r+1} shown in Figure 6. These tasks are ordered by precedence constraints because they share data. Thus, their release time r_i should follow the time sequence: scanning, detecting, way-point planning and then way-point communication with each barrel robot. This sequence can be modeled as a producer and consumer processing graph [8]. Using the techniques from [8], these tasks can be scheduled preemptively where priority ties between producer and consumer tasks in the processing graph of Figure 6 are broken in favor of the producer task. Schedulability tests in the later sections repeatedly exploit this result. The window resizing for the Hough transform is independent of other tasks. Further analysis shows that all periodic tasks may assume that their relative deadlines are equal to their periods:

$$p_i = d_i, i = 1, ..., 3 + N_r + 1.$$

The scanning task is the earliest task, so its release time r_s can be defined as the time origin, 0. The scanning task receives scanned position data of all barrel robots every 50ms. Thus, both period p_s and relative deadline d_s can be assigned 50ms. The execution time is dominated by the communication time between the laser scanner and the foreman. The communication speed C_s is set to be 500,000bps. The size of the position data for all N_r barrel robots surrounding the foreman is determined by hardware specification. However, considering the communication overhead, we set this size as $sData = 750$bytes. Thus, the WCET of the scanning task can be formulated:

$$e_s = \frac{sData}{C_s} = \frac{750 * 8}{500,000} = 12\text{ms}.$$

The detecting task also has a period of 50ms. Its release time follows the scanning task: $\phi_d = e_s$. The execution time is dominated by the refined Hough transform, whose running time is determined by three factors: the number of barrel robots N_r, the distribution of the barrel robots, and the number of sampling points S. As mentioned previously, we are only concerned with the WCET, which arises in the detecting task when the refined Hough transform degrades to its basic implementation. Thus, the WCET of the detecting task is a function of N_r and S. Let $e_d(N_r, S)$ be this WCET. By analyzing the implementation of the basic Hough transform and the resolution of the system clock, we model:

$$e_d(N_r, S) = C_1 N_r^2 + C_2 N_r + C_3 S + C4,$$

where C_1, C_2, C_3, C_4 are coefficients which are unknown and need to be determined from the foreman. We change N_r following the sequence, {2, 3, 5, 5} and S following the sequence, {26, 35, 47, 361} and thus get four independent linear equations:

$$\begin{cases} 4C_1 + 2C_2 + 26C_3 + C_4 = 12.93\text{ms} \\ 9C_1 + 3C_2 + 35C_3 + C_4 = 13.19\text{ms} \\ 25C_1 + 5C_2 + 47C_3 + C_4 = 13.81\text{ms} \\ 25C_1 + 5C_2 + 361C_3 + C_4 = 13.97\text{ms}. \end{cases}$$

Solving the above simultaneous linear equations yields:

$$\begin{aligned} C_1 &\approx 0.0172, \quad C_2 \approx 0.1695, \\ C_3 &\approx 0.0005, \quad \text{and} \quad C_4 \approx 12.5090. \end{aligned}$$

Thus, we estimate:

$$e_d(N_r, S) \approx 0.0172 N_r^2 + 0.1695 N_r + 0.0005 S + 12.5090\text{ms}. \quad (1)$$

A more sophisticated estimation may be derived using the multiple linear regression [16], which might improve the accuracy of the coefficients. However, more computational time is required, and not needed for this application.

The planning task is in charge of computing way-points for all N_r barrel robots. Its period and deadline are both 1,500ms. Its phase follows the execution of the scanning and detecting tasks, thus $\phi_p = e_s + e_d$. The WCET e_p is 16ms.

The way-point task set has a period and deadline of 1,500ms. Their phases $\{\phi_{wi} : i \leq N_r\}$ are set in terms of the precedence constraints. Execution time of a way-point task is determined by the communication time between the foreman and the barrel robots. Let C_w and $wData$ be the communication speed and the worst-case size of transmitted data respectively. In our system, they are 19,200bps and 20bytes. Thus, the execution time for each way-point task is:

$$e_w = \frac{wData}{C_w} = \frac{20 * 8}{19,200} = 8.33\text{ms}.$$

The window resizing task is used for the refined Hough transform. The basic idea is to dynamically change the search window for the Hough transform so that it can work with higher reliability and lower time complexity. (A detailed description was presented in Section 2.) Because this task is independent of other tasks with respect to precedence constraints, its phase ϕ_r can be set to 0. To enhance the performance of the Hough transform, the resizing period p_r is set to p_s. That means, whenever the new position data of barrel robots are collected, the search windows are updated correspondingly. Considering that this task is only for performance enhancement, the task deadline is soft. In other words, if this task can not be finished within d_r, windows for the Hough transform remain unchanged without fatal hazard to the system. One problem is that the task should guarantee that windows are not too stale.

Table 2 summarizes the parameter values for $N = 5$ barrel robots.

3.2 Static Task Scheduling under RM

Once we have modeled the tasks, the next important step is to design appropriate task scheduling approaches. Commonly used approaches to real-time system scheduling include clock-driven, static-priority driven and dynamic-priority driven. Clock driven (also called time driven) assigns time slots for specific tasks statically [1, 17]. In other words, the executing instants of all jobs are determined before jobs are released. Static-priority driven approaches schedule tasks in terms of their priorities, which are static for all jobs released by the same task. Priority can be determined by the task period. For instance, tasks with smaller periods have higher priority. This approach is called RM scheduling [20]. Priority can also be assigned according to the relative deadlines of tasks. A typical approach is deadline-monotonic (DM) [15], where tasks with smaller relative deadlines have higher priority. Like static-priority driven approaches, dynamic-priority driven approaches schedule tasks also in terms of their priorities. The difference, however, is that priority in the latter approaches change from job to job, even within the same task. Typical approaches include 1) earliest deadline first (EDF) [20], where jobs with earlier absolute deadline are assigned higher priority, and 2) least laxity first [15, 22], where jobs with less laxity time before their deadline have higher priority. Other scheduling approaches such as weighted round-robin are also presented in the real-time systems literature.

Correct selection of scheduling approaches may impact the predictability and resource utilization of real-time systems. In this application, the criterion for selecting the scheduling scheme is based on whether the task set of the system satisfies the sufficient schedulability conditions of the scheduling algorithm, so that barrel robots can be guaranteed to move smoothly. In this system, we present only one typical scheduling approach, rate-monotonic priority driven.

Sufficient schedulability tests of tasks under RM require first computing the system utilization [20, 4], which is defined as:

$$U = \sum_{i=1}^{n} \frac{e_i}{p_i},$$

where n is the number of tasks. Since the number of tasks and the execution time of the detecting task are characterized by the number of barrel robots N_r, utilization here is a function of N_r:

$$\begin{aligned} U(N_r) &= \sum_{i=1}^{n} \frac{e_i}{p_i} \\ &= \frac{e_s}{p_s} + \frac{e_d(N_r, S)}{p_d} + \frac{e_p}{p_p} + N_r \frac{e_w}{p_w} + \frac{e_r}{p_r} \\ &= 0.2907 + \frac{e_d(N_r, S)}{p_d} + 0.00556 * N_r. \end{aligned}$$

Setting $S = 361$, which is the default number of sampling points corresponding to the WCET of the detecting task, yields:

$$U(N_r) = 3.44 \times 10^{-4} N_r^2 + 0.0089 * N_r + 0.5445. \qquad (2)$$

Equation (2) shows that the system $U(N_r)$ approximates a quadratic function of N_r. Observing that the task set in the current design is simply periodic [20], the feasible schedulable condition is:

$$U(N_r) \leq 1.$$

Task ID	Task name	Phase ϕ_i (ms)	Period p_i (ms)	Execution time e_i (ms)	Deadline d_i (ms)
1	Scanning	0	50	12	50
2	Detecting	12	50	13.16	50
3	Planning	25.16	1500	16	1500
4	Way-point1	41.16	1500	8.33	1500
5	Way-point2	49.49	1500	8.33	1500
6	Way-point3	57.83	1500	8.33	1500
7	Way-point4	66.16	1500	8.33	1500
8	Way-point5	74.49	1500	8.33	1500
9	Window resizing	0	50	2	50

Table 2. Parameter values for 5 barrel robots.

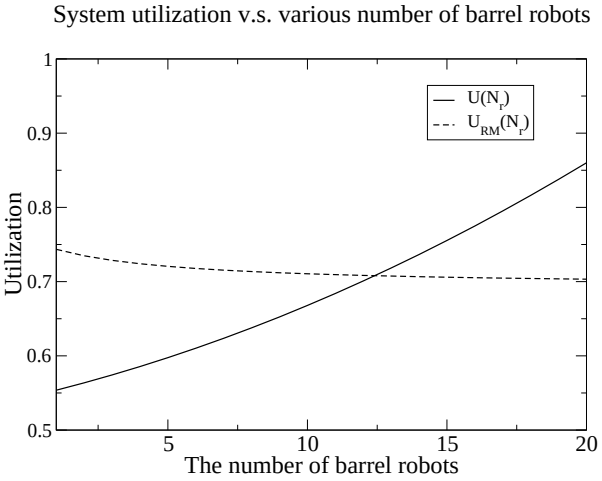


Figure 7. Solution to N_{max} in terms of $U_{RM}(N_r)$ and $U(N_r)$.

However, task sets in future scenarios may not be simply periodic. It would be meaningful to offer a general schedulable condition for our application. In [20], Liu and Layland show that Equation (3),

$$U_{RM}(n) = n(2^{1/n} - 1), \tag{3}$$

is a sufficient schedulable condition for RM scheduling of n independent, preemptable tasks with relative deadlines equal to their respective periods.

Here, $n = N_r + 4$. Thus, solving

$$U_{RM}(N_r + 4) = U(N_r)$$

gives the number of barrel robots, N_{max}, that can be controlled reliably by the foreman. The sufficient condition for the schedulability of the system under RM requires:

$$N_r \leq N_{max}.$$

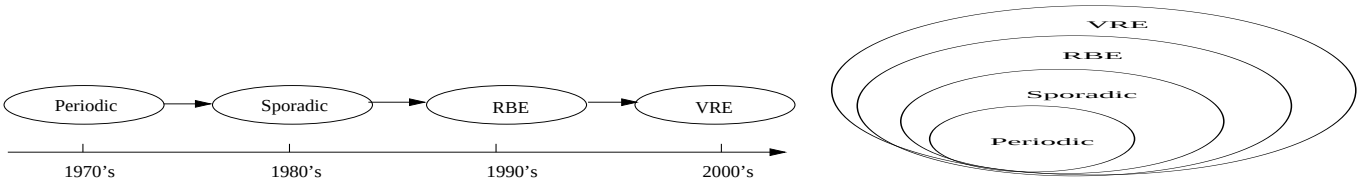
Figure 7 illustrates utilization curves of both functions $U_{RM}(N_r)$ and $U(N_r)$. Their intersection point is the solution of N_{max}, which is 12. Therefore, under RM scheduling approach, one foreman can safely control 12 barrel robots when the task set is not simply periodic. Please note that this conclusion is only a sufficient condition for RM scheduling. For a sufficient and necessary condition, we may use the time-demand analysis [14].

Thus, using the periodic task model and RM scheduling, the tasks can be guaranteed to receive their way-points in time. We can thus ensure smooth movement of barrel robots.

4 Enhanced Solution

Two canonical task models are the periodic model [20], where jobs are released periodically, and the sporadic model [22], where jobs are released with a minimum time separation. We have examined the simple periodic model in Section 3. A generalization to the sporadic model is *rate-based execution* (RBE) [10]. It makes no assumptions about the relationship between the points at which jobs are released for a task; it assumes that jobs are generated at a precise average rate but that the actual arrivals of jobs in time are arbitrary. This section presents the application of a more general task model, *variable rate execution* (VRE) [9]. The VRE task model provides two primary extensions to its predecessors: 1) variable WCET and periods, which may change at any time *t*, and 2) a dynamic task set in which tasks are allowed to enter and leave the system at arbitrary times. Summarily, Figure 8 illustrates task model evolution and their scopes by relaxing time constraints.

By careful analysis, we argue that tasks in our system are more suitably modeled as VRE than the simple periodic model discussed in Section 3. First, the number of way-point tasks are equal to the the number of barrel robots. This means whenever a barrel robot moves beyond the scannable region of the foreman, its corresponding way-point task should be removed from the task set. Similarly, whenever a new barrel robot is added to the group, the system should create its corresponding way-point task. These events are detected by the refined Hough transform. Second, WCET of the detecting task is subject to the number of the barrel robots N_r. Equation (1) shows that the WCET of the detecting task is quadratically proportional to N_r. Thus, the VRE



(A) Evolution (B) Scope

Figure 8. Evolution of task models (left) and their scopes (right).

is a more appropriate task model.

Following the notion in [9], a VRE task T_i is described by 4-tuple $(x_i(t), y_i(t), c_i(t), d_i(t))$, where:

- $y_i(t)$ is an interval or period in time,
- $x_i(t)$ is the number of jobs expected to be released in $y_i(t)$,
- $c_i(t)$ is the WCET, and
- $d_i(t)$ is the relative deadline, which is typically equal to $y_i(t)$. In the paper, we thus assume $y_i(t) = d_i(t), \forall i, t$.

Each parameter is a function of time t. Let the jth job of T_i be J_{ij}. Its absolute deadline $D_i(j)$ is then assigned as follows:

$$D_i(j) = \begin{cases} t_{ij} + d_i(t) & \text{if } 1 \leq j \leq x_i(t) \\ max\Big(t_{ij} + d_i(t), \\ D_i(j - x_i(t)) + y_i(t)\Big) & \text{if } j > x_i(t), \end{cases}$$

where t_{ij} is the release time of job J_{ij}. This formulation implies two practical properties: 1) up to $x_i(t)$ jobs may contend for the processor with the same deadline and 2) deadlines of jobs J_{ij} and $J_{i,j+x_i(t)}$ are separated by at least $y_i(t)$ time unit.

For analysis convenience, our enhanced solution assumes that 1) the system task set $\{(x_i(t), y_i(t), c_i(t), d_i(t))\}$ is subject to only $N_r(t)$, the variable number of the barrel robots at arbitrary times; 2) the period values of all tasks remain unchanged as presented in Section 3; and 3) S is set to be the default value, 361. Thus, the dynamic parameters existing in the enhanced solution are $c_i(t)$ and the number of tasks. Substituting $N_r(t)$ into Equation (2) gives

$$U(N_r(t)) = 3.44 \times 10^{-4} N_r(t)^2 + 0.0089 * N_r(t) + 0.5445,$$

which is the variable system utilization under the dynamic task set:

$$V(t) = \{T_i : i = 1, .., N_r(t) + 4\}.$$

Thus, in terms of the conclusion shown in [9], the sufficient condition for the EDF scheduling approach is

$$\forall t, \sum_{i \in V(t)} f_i(t) \leq 1, \tag{4}$$

where

$$\sum_{i \in V(t)} f_i(t) = U(N_r(t)).$$

The VRE model presented in [9] provides the theoretical foundations for scheduling VRE tasks. However, the actual execution of tasks based on the VRE model requires a VRE-EDF scheduler that assigns deadlines in accordance with the model. While such a scheduler has been implemented in Linux and a modified version of the μC/OS-II (MicroC/OS-II) [19] real time operating system, most operating systems do not support EDF scheduling—let alone VRE-EDF scheduling.

Fortunately, the attributes of the task set in this system result in RM scheduling producing the exact same schedule that would be produced under VRE-EDF scheduling where deadline ties are broken in favor of producer nodes in the processing graph shown in Figure 6. Thus, the same scheduling algorithm used for the basic, static task model solution can be used in the enhanced, dynamic solution. As long as WCET changes are only made for tasks at the end of their execution period and $\forall t, \sum_{i \in V(t)} f_i(t) \leq 1$, the system performs correctly.

5 Conclusion

In this paper we present the design and implementation of a real-time model for the robotic highway safety marker system. We extensively analyze the current system and model the periodic tasks with a 4-tuple (ϕ, p, e, d). Periodic tasks in the system include scanning, detecting, planning, N_r waypoints, and window resizing.

To decrease the time complexity and increase the reliability of barrel robot detecting in the system, we present a refined implementation of the Hough transform. Its basic idea is to limit the search window for barrel robots. Results show that this refined Hough transform outperforms the traditional implementation in both time complexity and reliability with only slight additional storage for windows.

We then present a typical real-time approach using rate-monotonic scheduling. Analysis shows that the current system can be feasibly scheduled. This conclusion justifies that the path for each barrel robot can be guaranteed to be smooth under the designed real-time system. For the scalability issue, we offer a sufficient condition for the upper bound on the number of barrel robots schedulable under RM.

Using the VRE model, this paper presents an enhanced solution that supports dynamic changes to the number of barrel robots and the WCET of the detecting task. That is, barrel robots may enter or leave the control from one foreman to another foreman at arbitrary times in the working field. The scalability is thus improved and cost of the system is reduced. The enhanced system is schedulable under either RM or EDF if Condition (4) holds.

Future work for this system includes:

- relax the constraints imposed on the designed task model, such as resource sharing;
- introduce slack stealing techniques to decrease the response time of aperiodic tasks.

Acknowledgment

The authors would like to thank Janson Dumpert for his help on parameter computations.

References

[1] T. P. Baker and A. Shaw, "The Cyclic Executive Model and Ada," *Proceedings of Real-Time Systems Symposium*, pp.120-129, 1988.

[2] J. Borestein and Y. Koren, "The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots," *IEEE Trans. on Robotics and Automation*, Vol.3, No.3, 1991.

[3] R. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE Trans. on Robotics and Automation*, Vol.2, No.1, 1986.

[4] U. C. Devi, "An Improved Schedulability Test for Uniprocessor Periodic Task Systems," *Proceedings of the 15th Euromicro Conference on Real-Time Systems*, pp. 2003.

[5] R. O. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *In Communications of the ACM*, pages 11-15, 1972.

[6] S. Farritor, H. Hacot, and S. Dubowsky, "Physics-based Planning for Planetary Exploration," *IEEE Intl. Conference on Robotic and Automation*,1998.

[7] S. M. Farritor and Mark E Rentschier, "Robotic Highway Safety Markers," *In Chris Mellish, editor, ASME International Mechanical Engineering Congress and Exposition*, Montreal, 2002.

[8] S. Goddard and K. Jeffay, "Managing Latency and Buffer Requirements in Processing Graph Chains," *The Computer Journal*, 44-6:486-503, 2001.

[9] S. Goddard and X. Liu, "A Variable Rate Execution Model," Technical Report TR-UNL-CSE-2003-15 (in submission to ECRTS'04), Department of Computer Science and Engineering, University of Nebraska - Lincoln, December 2003.

[10] K. Jeffay and S. Goddard, "A Theory of Rate-based Execution," *Proceedings of the 20th IEEE Real-Time Systems Symposium*, Phoenix, Arizona, December 1999, pp.304-314.

[11] O. Khatib, "Real-Iime Obstacle Avoidance for Manipulators and Mobile Robots," *Intl. Journal of Robotics Research*, 5(1), 90-98,1986.

[12] T. Lasky and B. Ravani, "Sensor-based Path Planning and Motion Control for a Robotic System for Roadway Crack Sealing," *IEEE Trans. on Control Systems Tech.*, Vol. 8, No.4, pp.609-622, 2000.

[13] H. M. Lee, J.Kittle, and K.C.Wong, "Generalized Hough Transform In Object Recognition," *11th IAPR international conference*, Vol. III. pp. 245-289, 1992.

[14] J. P. Lehoczky, L. Sha, and Y.Ding, "The Rate-Monotonic Scheduling algorithm: Exact Characterization and Average Case Behavior," *Proceedings of Real-Time Systems Symposium*, Decemember 1989, pp.166-171.

[15] J.Y.T. Leung and J.Whitehead, "On the Complexity of Fixed-Priority Scheduling of Periodic, Real-Time Tasks," *In Performance Evaluation*, 1982.

[16] D. M. Levine, P. P. Ramsey, and R. K. Smidt, "Applied Statistics", pp.567-pp.670, Prentice Hall, 2001.

[17] J. Liu, "Real-Time Systems",Prentice Hall, 2000.

[18] X. Liu and S. Goddard, "A Loadable Variable Rate Execution Scheduler," *Proceedings of the Real-Time Linux Workshop*, Valencia, Spain, November 2003, pp.187-196.

[19] J. Labrosse. *The Real Time Kernel MicroC/OS-II*, CMP Books, May 2002.

[20] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment," *Journal of the ACM (JACM)*, 20(1):46-61, 1973.

[21] A. Mohri, M. Yamamoto, and S. Marushima, "Collision-Free Trajectory Planning for Two Manipulators Using virtual Coordination Space,", *IEEE Intl. Conference on Robotics and Automation*, pp.674-679,1993.

[22] A. K. Mok,"Fundamental Design Problems of Distributed Systems for the Hard Real-Time Environment," Ph.D. Thesis,MIT, Department of EE and CS, MIT/LCS/TR-297, May 1983.

[23] S. Shah, "New Work Zone Safety Devices," *Proceedings of the ASCE 3rd Intl. Conference on Applications of Advanced Technologies in Transportation Engineering*, pp.308-315, 1996.

[24] L. G. Shapiro and G.C. Stockman, "Computer Vision, Chapter 10," Prentice Hall, NJ, 2001.

[25] X. Shen, "Control of robotic highway safety markers," M.S. thesis, Computer Science and Engineering, University of Nebraska-Lincoln, 2003.

[26] M.Sonka, V. Hlavac, and R. Boyle, "Image Processing: Analysis and Machine Vision, 2nd edition," Chapman & Hall Computing, pp.679-722, 1998.

[27] S. A. Velinsky, "Heavy Vehicle System for Automated Pavement Crack Sealing," *Heavy Vechile Systems, Intl. Journal of Vehicle Design*, Vol.1, No.1, pp.114-128, 1993.

[28] T. H. West, S. A. Velinsky, and B. Ravani, "Advanced Highway Maintenance and Construction Technology Applications," *TR news*, pp.17-23, 1995.

[29] T. White, "Evolving Automation in the Asphalt Paving Industry," *TR news*, No.176, pp.4-6, 1995.

[30] D. Woo, "Robotics in Highway Construction," *Public Roads*, Vol.58, No.3, pp.26-30, 1995.