# A Benchmark Suite for Mobile Robots

Jacky Baltes
Centre for Imaging Technology and Robotics
University of Auckland
Private Bag 92019, Auckland 1, New Zealand
j.baltes@auckland.ac.nz

**Abstract**

*This paper describes a benchmark suite for mobile robots that provides quantitative measurements of a mobile robot's ability to perform specific tasks. Guidelines for the design of benchmark tests were derived from other areas faced with the problem of evaluating complex systems. The benchmarks test the control and accuracy of the path and trajectory tracking, the static path planning, and the dynamic path planning ability of a mobile robot. A set of metrics that provide important information about a mobile robot's performance are also presented. These benchmarks could also be used as simple games. Their inclusion in robotic games will lead to an increased opportunity fo researchers to evaluate their work without having to buy expensive or special purpose equipment.*

## 1 Introduction

Immediately after the first robots appeared in research labs and universities, they were used for games and the first robotic competitions were organized. An example is the table tennis playing robot developed at MIT in the late 60's.

Robotic games have been used for a variety of reasons in robotics research:

They included problems that both posed interesting research questions as well as were motivating. Although many of the associated problems are similar, it is easier to convince people to work on a treasure collecting robot than on a vacuuming robot.

They are an intuitive way to introduce robotics and AI research to the general public. This has lead to a strong interest in robotic games in the media and associated with this an increased publicity.

They sometimes allow researchers to secure additional funding through the increased publicity.

They provide researchers with a venue to show their work, and discuss ideas in an informal setting.

Another reason often cited for the use of robotics games is that they allow an empirical evaluation (benchmark) of a team's research and their progress. However, as a competition matures, this becomes increasingly more difficult, since more expensive and more special purpose hardware is being developed for the competition. There is also the ugly side of these competitions where teams start to exploit the rules. Winning the competition becomes the over-ruling motivation.

Other researchers have noticed this trend as well. Kitano suggests a set of physical agent challenges for the RoboCup domain. However, these challenges (e.g., receiving a pass) are high level tasks, difficult to reproduce, limited to specific robot hardware, and do not allow to isolate the performance of different subsystems.

This paper gives a brief introduction to popular robotic games (Section 2) as well as a background in the design of benchmarks (Section 3). From this background work, a set of simple benchmarks and their associated performance measures are described (Section 4). These benchmarks have been used with good success in our research work. The motivation is that these benchmarks allow a team to gauge their progress, even with limited hardware and no special purpose equipment.

## 2 Robotic Games

This section describes some of the more well known robotics games: Micromouse, Robocup, and RoboFesta.

### 2.1 Micromouse

The Micromouse competition, first suggested by Don Christiansen [3] was the first robotics competi-

tion that elicited large interest in the media.

The playing field consists of a maze constructed from small plastic walls. A robot is first allowed to explore the maze and to find the shortest path from the initial position to the goal. After the robot completes its exploration, it is returned to the starting position and has to move to the goal position as quickly as possible.

Even after 23 years, Micromouse competitions are still popular. However, most researchers now consider the AI and robotics problems as solved. Nowadays, most improvements are mechanical in nature and result in lighter and faster robots[2].

Also, a number of researchers have adapted and extended the original Micromouse competition. For example, in Singapore, the Micromouse has developed into a local trash bin collection competition where robots need to collect trash bins.

### 2.2 RoboCup

Robots playing soccer was first posed as a challenge problem to the AI community by Alan Mackworth in 1992 [6]. The idea was taken up by Hiroaki Kitano who organized the first international competitions in 1996.

Playing soccer adds two important dimensions to robotic games. Firstly, instead of a single agent, a successful soccer team has to coordinate the actions of all robots in the team. Secondly, a soccer game has an active opponent that is trying to prevent a robot from executing its plan.

The RoboCup initiative currently has three active leagues: simulator, small sized, and medium sized.

The small sized league is intended as the entry level competition for physical robots. Teams consist of five players and the playing field is the size of a table tennis plate. Teams can use a global vision system. This is the most active physical robot league; 28 teams registered for the RoboCup 2000. Most teams use holonomic robots with a wheel chair based design, an embedded microcoontroller, and a global vision camera mounted directly overhead. Coloured markers are used to determine the position, orientation, and id of robots. However, some teams made a conscious decision to enter the competition with inferior hardware or additional constraints. For example, some teams use local vision (e.g., CHIIPS Glory from UWA), nonholonomic robots (All Botz). Some legged and humanoid robot teams have expressed interest in competing in this league.

The trend towards multi-purpose robots is important and supported by robot games that include a variety of competitions. It is a natural counter agent towards over-specialization.

### 2.3 RoboFesta

RoboFesta is possibly the most ambitious robotics competition, the Olympics of robot games. The first competition is planned to last for 47 days with events being organized in five different sites. Also, it is planned that the competition will include around 30 different games with over 7000 competitors. The games include well established competitions such as RoboCup and the Micromouse competition discussed in the previous subsections as well as new ones.

RoboFesta is the first attempt to cover the full range of work in robotics, from simple introductory single robot games to complex tasks for teams of robots, from small robots to the very large ones, and from land-based to aquatic and air-borne robots.

## 3 Benchmarks

Many benchmarks have a bad reputation in the research community because they are often (mis)-used for marketing purposes. However, there are also some legitimate reasons for using benchmarks. Most importantly, to evaluate progress in the field quantifiable measures are needed. A well designed set of benchmarks that accurately reflects real world usage patterns can direct research and highlight deficiencies.

### 3.1 Cars

When buying a car, one quickly finds out that there is a staggering variety of different makes and models: family station wagons, the 4x4 jeeps, exotic sports cars. In spite of this variety, there are a set of commonly used measurements, such as maximum speed, acceleration from 0 to 100 km/h, fuel consumption in city traffic, that are used by buyers to rate and compare cars.

In most applications, these numbers would present little useful information for mobile robots. For example, a robot's maximum speed and the top speed at which the robot can be controlled safely are usually two very different speeds. The latter speed is probably more important than the former for a user of a mobile robot. But "to control safely" is not precisely defined and depends on the current situation.

### 3.2 Processor Benchmarks

Researchers working on computer architecture and organization are also faced with the problem of accurately measuring the performance of a complex hardware/software system.

A very early method of evaluating processor performance were "millions of instructions per second (MIPS)" and "millions of floating point operations per second (MFLOPS)" ratings. These were used commonly until the late 1980's. However, once RISC processors appeared on the market, the main weakness of these ratings became readily apparent; instructions and floating point operations are not clearly defined.

Equivalent measures for a mobile robot systems are for example the video frame rate, the control cycle time, or the path planning time. However, the problem is similar to that of MIPS ratings. For example, the function of the video processor is not precisely defined. Most video servers in the RoboCup return information about the position and orientation of objects in the domain, but some also compute their velocities, errors from the desired positions etc.

Researchers realized that processor performance is determined by three factors: the number of instructions, the average clocks per instructions, and the clock frequency. Trying to evaluate performance using a subset of these features leads to meaningless results. Processors must be evaluated using real world applications.

Early popular benchmark programs were small toy programs such as the popular Dhrystones and Wheatstones benchmarks. The fact that these programs were easy to understand and their behavior easy to analyze lead to some people exploiting the benchmarks for marketing purposes. For example, DEC used a C compiler with a special DHRYSTONE flag. This flag would turn on some optimizations in the compiler which in general would reduce the efficiency of the generated code, but would improve performance dramatically on the Dhrystone benchmark.

These shortcomings led a number of companies to form the SPEC group in 1999. The SPEC CPU benchmark consists of parts from eight real applications ranging from Neural Net simulators to the GNU C compiler[4].

### 3.3 AI Benchmark Problems

For a long time, there has been little quantitative and comparative evaluation in AI research in general. Since the real world version of the problems are often too difficult, researchers have often used synthetic problems or domains. When doing robotics research, the actual hardware was abstracted and the problem was solved in a computer simulation. Often, these problems, domains, or simulation environments were created by the designer of the program to be tested. It is thus hardly surprising that relatively few failures of AI systems have been reported. Since the designer can be god in the simulated world, she can make the world fit the program. Even when trying her best to design a fair simulation environment, it is hard to avoid implicit assumptions creeping into the design of the problem as well as the solution.

Comparing AI systems on a test suit is difficult since representation languages for the problems, domains, simulation were incompatible and since these representations could have a dramatic effect on the performance of an AI system.

A few simple toy domains such as the blocks world (stacking and unstacking of blocks), the Towers of Hanoi, or the 15 piece sliding puzzle have been used in AI. However, the representation of a domain can greatly influence the performance of a system and this representation was not standardized. A clver designer can encode a lot of information about the problem in the representation. For example, a planner in the blocks world can be sped up by several orders of magnitude if the planner is told that a block can never be on top of itself.

More recently, a few communities in AI have created a set of benchmark problems that are now used commonly in quantifying research results. The first such community was the machine learning community that created the UCSD ML benchmark problems [7].

It is obviously important that benchmark problems are representative of problems in the real world. For example, Holte showed that most of the problems in the UCSD machine learning benchmarks were very biased [5]. He showed that even naive learning algorithm could achieve more than 90% accuracy on most problems in the dataset. Most problems in the real world are of course much more complex.

## 4 Mobile Robot Benchmarks

In contrast to cars and processors, mobile robots do not (yet) have the advantage of a large user and application base. Therefore, the design of useful benchmarks is even more important. A strong user base can identify weaknesses in the benchmarks through anomalies between benchmark and real world results.

Using the lessons learned from the examples given above, a number of guidelines for the design of mobile

robots benchmarks can be extrapolated:

A benchmark is a performance measurement with respect to a particular task. Any benchmark that is based on only subparts can be misleading. So a benchmark problem for mobile robots should include the full sensing → perception → reasoning → acting cycle.

Benchmarks must be portable. A large variety of different mobile robots should be able to execute them. In particular, benchmarks that require expensive hardware must be avoided. For example, holonomic wheeled robots, non-holonomic wheeled robots, legged robots, and humanoid robots with different physical dimensions and different sensors must be able to perform these benchmark tests.

The quantitative metrics of the benchmarks must be easily observable without any detailed knowledge of the underlying architecture. Therefore, the SPEC benchmarks use execution time rather than number of cache misses as metrics. For example, some architectures may not have an explicit representations of a path, so it is impossible to test the speed of its path planning component.

Benchmarks must be updated. The basic underlying assumption is that these tasks reflect real world applications. However, since new applications will be developed, any benchmark must be seen as temporary and must be amenable to adaptation and improvements in the future.

This section will suggest a number of benchmarks for mobile robots suitable to the current state of the art in hardware and software. There are a number of subproblems in mobile robotics, such as localization, path planning, and control, that must somehow be addressed by the designer. The goal of the benchmarks described in the following subsections is to test the individual subparts as well as their combination.

### 4.1 Path and Trajectory Following

A funadmental problem for any mobile robot is movement. The benchmark designed in this subsection is being used by our group to compare the performance of different path following or trajectory following algorithms. The path following benchmark uses a path where all measurements are relative to the size of the robot and its maximum turn radius.

Some of the robots used a global vision system to control the robots, but this setup is also used to evaluate the performance of local vision robots.

Since the aim of this benchmark is to evaluate the path following ability of the robot, the environment is not specified. For example, when evaluating our local vision robots, the path is marked on the floor using a white line and suitable coloured markers are dispersed over the environment to provide local and global position and orientation information. Note that even in purely reactive systems, where there may not even exist an explicit representation of a path (e.g., subsumption architectures), the robot can be controlled by providing the correct set of stimuli in the environment.

A global vision system is mounted overhead and collects information about the path or trajectory tracking for later analysis. The analysis is influenced by the accuracy of the position information and by the frame rate of the video-server. The video-server that we designed is able to provide position information with less than 3cm position error over a large area (25 $m^2$)[1] at 50 updates/second, which is sufficient.

The global vision server observes the progress of the robot. The metric reported for this benchmark is the average and maximum position error as well as the timing errors. Orientation errors are not listed, since they may or may not be important. For example, for commonly used wheel chair robots, orientation errors are not important since these robots can turn on the spot. In the case of car-like robots, large orientation errors lead automatically to large position errors.

We used this benchmark on three different mobile robot platforms. The length of the robots were 15cm, 18cm, and 40cm respectively. The majority of the robots used global vision, but some of the robots used local vision.

This benchmark has been used to evaluate the performance of different control algorithms for car-like mobile robots: three different bang bang controllers, a Fuzzy Logic controller, a reinforcement learning controller, and a look ahead controller. For example, Fig. 1 shows the path followed by the reinforcement learner controller using the 18cm robots and a global vision system.

### 4.2 Static Path Planning

For a mobile robot to be useful, it must be in the right position at the right time. This requires the robot to create a path from its current position to the goal position, so called path planning.

However, there are many different versions of the path planning problem. Some path planning algorithms use complete knowledge of the environment and assume that obstacles do not move. Other planning methods assume only local knowledge and deal

[1] The position error for smaller playing areas is significantly smaller. For the RoboCup playing field (2.74m by 1.52m), the error is less than 1cm

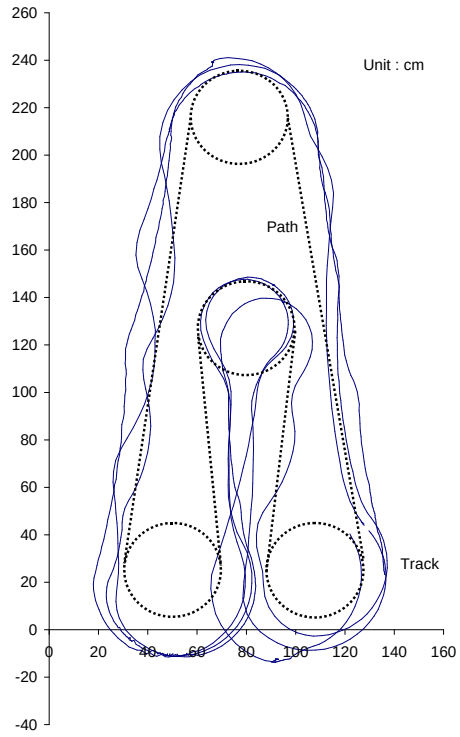


Figure 1: Path of the Reinforcement Learning Controller

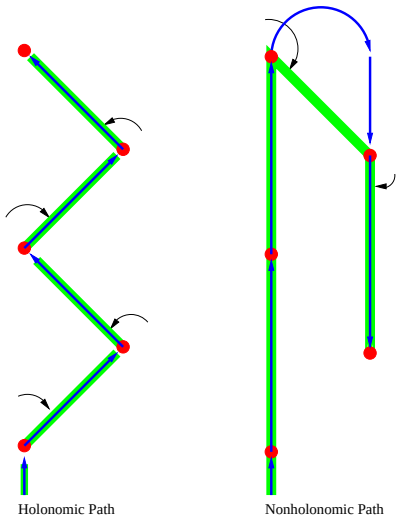


Figure 2: Simple Path Planner Example. The figure shows the actual path and the approximated holonomic path.

with dynamic environments. This great variability in path planners makes it difficult to design a benchmark that is applicable to all mobile robots.

The benchmark described in this subsection, therefore, limits itself to determining a robot's ability to reach a number of desired goal positions. The robot must indicate when it reaches a goal position. For example, a robot can flash an LED or simply stop for a few seconds before proceeding to the next target.

In the absence of any obstacles, a simple greedy path planner is able to create the shortest path for a holonomic robot. However, given the dynamics of the robot, this path may not be the quickest path. A simple example is shown in Fig. 2. The first path is shorter, but requires many more turns. The second path is more suitable for a car-like robot.

Developing a metric to measure the performance of a path planner is non-trivial. Simple metrics such as the time to reach all goal locations, and the total distance traveled are in large part determined by the controller rather than the path planner. For example, Fig. 3 is a trace of Balluchi's controller following a given path. Even for the straight line segment in the lower right, it would be difficult to infer that the robot is following a straight line. But, we do want to factor out or at least minimize the influence of the controller in this benchmark.

The general problem is one of plan deduction. Given a trace of the observations from the robot's behavior, we have to infer the plan that the robot was trying to follow. Possible path segments for a robot are undefined. For example, most path planners for wheelchair robots contain only straight lines and turns on the spot. For most car-like robots, paths consist of straight lines and maximum turns. Other path planners based on splines or potential field have a larger set of possible path segments.

Since straight lines are path segments available to most mobile robots, we use straight lines and changes in orientation to deduce the plan of the robot. This is clearly an approximation, but assuming that the goals are not too close, this error will not be significant.

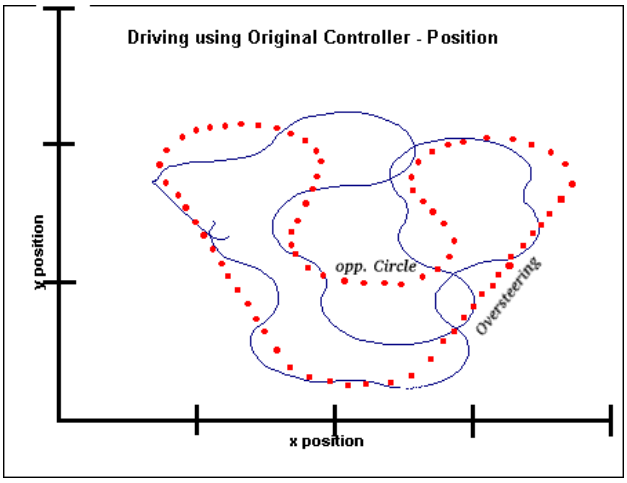


Figure 3: Balluchi's Controller Following a Simple Path. It is difficult to infer the path (dots) from the shown track.

Figure 2 shows the inferred plan for the two sample paths shown.

Also note that our system deduces a plan for the robot's behavior, even if the robot itself does not have an explicit representation of a plan, since it is based on the observed behavior of the robot solely.

We have the static path planner benchmark with four different path planners: Bicchi's path planner [1], an optimized planner for the RoboCup domain, an adaptive planner, and an anytime planner.

### 4.3 Dynamic Path Planning

The task described above evaluates the performance of the path planning component of a mobile robot in static domains. However, all robots must deal with the real world, which is dynamic and uncertain. Therefore, we are currently working on the design of a new benchmark for dynamic path planners.

The dynamic path planning problem is similar to the path planning problem described in subsection 4.2 and is a game of catch between two robots.

The first robot, called the evader, is the dynamic goal location and follows a predefined path. There are many different possible path for the evader; for our first experiments, we choose a simple oval shaped path. The task of the robot, called the pursuer, is to catch the evader.

The performance metric used for these benchmarks is the total time that it takes the pursuer to catch the evader. A better metric would be the number of times that the robot changed its plan. However, from observation alone it is difficult to infer this information, which is why this benchmark only provides a summary score.

## 5 Conclusion

Although extremely popular and entertaining, robot games by themselves do not allow to evaluate research progress. Results such as 33:0 do not tell the complete story. Quantitative performance measures for different components of a mobile robot are needed.

This paper compares a number of different approaches that have been used in other fields to evaluate the performance of complex hard and software systems. Based on the lessons learned in these fields, a set of benchmark problems has been proposed: control, static path planning, and dynamic path planning.

The current benchmarks should be seen as a first step towards more quantifiable results at robot competitions.

## References

[1] Antonio Bicchi, Giuseppe Casalino, and Corrado Santilli. Planning shortest bounded-curvature paths for a class of nonholomic vehicles among obstacles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1349–1354, 1995.

[2] Thomas Bräunl. Research relevance of mobile robot competitions. *IEEE Robotics and Automation Magazine*, 6(4):32–37, December 1999.

[3] Donald Christiansen. Announcing the amazing micromouse maze contest. *IEEE Spectrum*, 14(5):27, May 1977.

[4] The SPEC Group. The spec cpu 2000 benchmark. WWW: http://www.specbench.org, February 2000.

[5] Robert Holte. Very simple classification rules perform well on most data sets. Technical Report TR-91-16, University of Ottawa, 1991.

[6] Alan Mackworth. *Computer Vision: System, Theory, and Applications*, chapter 1, pages 1–13. World Scientific Press, Singapore, 1993.

[7] C.J. Merz and P.M. Murphy. UCI repository of machine learning databases, 1998.