
**Lecture Notes for
IROS 2006 Workshop II (WS-2)
Tuesday, October 10, 2006**

Benchmarks in Robotics Research

**Organizer
Angel P. del Pobil
Universitat Jaume I
pobil@uji.es**



Rationale

Current practice of publishing research results in robotics makes it extremely difficult not only to compare results of different approaches, but also to assess the quality of the research presented by the authors. Though for pure theoretical articles this may not be the case, typically when researchers claim that their particular algorithm or system is capable of achieving some performance, those claims are intrinsically unverifiable, either because it is their unique system or just because a lack of experimental details, including working hypothesis. This is, of course, partly due to the very nature of robotics research: reported results are tested by solving a limited set of specific examples on different types of scenarios, using different underlying software libraries, incompatible problem representations, and implemented by different people using different hardware, including computers, sensors, arms, grippers...

This state of affairs cannot be changed in the short term, but some steps can be taken in the right direction by studying the ways in which research results in robotics can be assessed and compared. In this context, the *European Robotics Research Network* EURON has as one of its major goals the definition and promotion of benchmarks for robotics research. A successful workshop on this topic was organized last March at Palermo in conjunction with EUROS'06, the *First European Symposium on Robotics*. The main purpose of this second workshop is to address these issues by providing an informal forum for participants to exchange their on-going work and ideas in this regard. It is intended as a forum for discussion, exchange of points of view, assessment of results and methods, and as a source of dissemination and promotion in the area of benchmarks in robotics research.

Authors (in alphabetical order)

- S. Behnke, University of Freiburg, Germany. behnke@informatik.uni-freiburg.de
- A. Bonarini, Politecnico di Milano, Italy. bonarini@elet.polimi.it
- W. Burgard, University of Freiburg, Germany. burgard@informatik.uni-freiburg.de
- E. Cervera, Universitat Jaume I, Spain. ecervera@icc.uji.es
- G. Fontana, Politecnico di Milano, Italy. fontana@elet.polimi.it
- R. Geraerts, Utrecht University, the Netherlands. roland@cs.uu.nl
- P. Jensfelt, Royal Institute of Technology, Sweden. patric@kth.se
- D. Kragic, Royal Institute of Technology, Sweden. dani@kth.se
- V. Kyrki, Lappeenranta University of Technology, Finland. kyrki@lut.fi
- F. Lingelbach, Royal Institute of Technology, Sweden. frankl@kth.se
- M. Matteucci, Politecnico di Milano, Italy. matteucci@elet.polimi.it
- J. Minguez, Universidad de Zaragoza, Spain. jminguez@unizar.es
- Morales, Universitat Jaume I, Spain. morales@icc.uji.es
- P. del Pobil, Universitat Jaume I, Spain. pobil@uji.es
- Rano, Universidad de Zaragoza, Spain. irano@unizar.es
- P.J. Sanz, Universitat Jaume I, Spain. sanzp@icc.uji.es
- D. G. Sorrenti, Univ. Milano-Bicocca, Italy. sorrenti@disco.unimib.it
- J. D. Tardos, Universidad de Zaragoza, Spain. tardos@unizar.es

Program

Why do We Need Benchmarks in Robotics Research?

Angel P. del Pôbil

Robot Competitions - Ideal Benchmarks for Robotics

Sven Behnke

RAWSEEDS: Robotics Advancement through Web-publishing of Sensorial and Elaborated Extensive Data Sets

A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti and J. D. Tardos

Steps Towards the Automatic Evaluation of Robot Obstacle Avoidance Algorithms

Ignacio Rañó and Javier Mínguez

On Experimental Research in Sampling-based Motion Planning

Roland Geraerts

Cross-Platform Software for Benchmarks on Visual Servoing

Enric Cervera

Drawing a parallel: Benchmarking in Computer Vision and Robotics

Danica Kragic, Ville Kyrki, Patric Jensfelt and Frank Lingelbach

Experiments in Visually-Guided 3-Finger Grasping

Antonio Morales, Pedro J. Sanz and Angel P. del Pobil

Why do We Need Benchmarks in Robotics Research?

Angel P. del Pobil
Robotic Intelligence Laboratory
Universitat Jaume I
Castellón, Spain
pobil@uji.es

Abstract - *This paper discusses the current state of affairs in measuring and comparing research results in robotics and describes current actions towards better benchmarking practice in the field.*

1 State of the art

It is a well-known fact that the current practice of publishing research results in robotics makes it extremely difficult not only to compare results of different approaches, but also to assess the quality of the research presented by the authors. Though for pure theoretical articles this may not be the case, typically when researchers claim that their particular algorithm or system is capable of achieving some performance, those claims are intrinsically unverifiable, either because it is their unique system or just because a lack of experimental details, including working hypothesis. Often papers published in robotics journals and generally considered as good would not meet the minimum requirements in domains in which good practice calls for the inclusion of a detailed section describing the materials and experimental methods that support the authors' claims.

Moreover, the problems claimed to be solved in this kind of papers become the *state of the art*, so that more solid papers supported by exhaustive experiments are rejected because the problem is considered to *be already solved*, typically referring to one of those papers. Even though this is partly due to the very nature of robotics research: reported results are tested by solving a limited set of specific examples on different types of scenarios, using different underlying software libraries, incompatible problem representations, and implemented by different people using different hardware, including computers, sensors, arms, grippers... this situation is impeding solid progress in the field and jeopardizing the credibility of robotics research.

This state of affairs cannot be changed in the short term, but some steps can be taken in the right direction by studying the ways in which research results in robotics can be assessed and compared. In this context networks and societies, such as EURON or IEEE, can play an important role. The long-term benefits of systematic benchmarking

practice are evident: not only will it foster the overall quality of research results but it will also improve publication opportunities for solid research, thereby increasing international visibility of such research and lead to rapid adoption of new research results by application developers and the robotics industry.

1.1 Robot competitions

Curiously enough, the above described situation is compatible with the fact that some of the most popular organized events in robotics are related to comparative research: the different successful robot competitions that exist today are a way of comparing the performance of the competing systems by means of very well-defined rules and metrics. The organization of these scientific competitions has proven a quick way to attract substantial research efforts and rapid to produce high-quality working solutions.

1.2 Task Forces and Grand Challenges

When considering the role of societies and networks in relation to these issues, trying to set up a task force to define a set of *gold standards* in robotics by itself does not seem a feasible approach. To mention just a recent example: DARPA and NSF funded a study about a very particular field in robotics, namely human robot interaction (HRI). Even for this reduced field over sixty representatives from academia, government and industry participated in the study, and one of the recommendations regarding actions for the next 5 years concluded that *the HRI field is still too new to set milestones or benchmarks* [1]. Even so, some grand challenges were proposed. Grand challenges are interesting as long-term goals, but they are usually vaguely described, resulting from a roadmap in the field, and not very useful for measuring progress or comparing results. Nevertheless, benchmarks could be conceived as a way of measuring progress toward a grand challenge.

2 Methodology

Defining a benchmark—even a sound valid benchmark—could be an easy task, if it is just taken as an academic exercise. Defining a successful benchmark is

something completely different. A benchmark can be considered as successful if the community extensively uses it in publications, conferences and reports as a way of measuring and comparing results. To put it in a few words: a benchmark is successful if and only if it is widely accepted by the community at which it is targeted.

This kind of success is somehow difficult to predict, but some the following considerations may help:

- its success is related to its quality, i.e. it really must serve its purpose
- robotics is a too broad field and defining benchmarks for robotics in general does not seem to make sense, but rather benchmarks should focus on particular sub-domains, such as visual servoing, grasping, motion planning, ...
- it is easier that a benchmark is successful within a scientific (sub)community if it arises from the community itself instead of having it defined outside this community.

Reaching consensus does take time: proposing good benchmarks for the community to accept, is a long process that requires the concourse of many people in many subcommunities within robotics. Consequently, the role of organizations in this context does not seem to be that of defining benchmarks, but rather to propose and encourage:

- by convening the different sub-communities within robotics so that they become aware of the problem, and encourage them to make efforts to define these benchmarks, follow their work, help in disseminating their proposals so that they become publicly known.
- by helping in providing the required information about other efforts in the field, organizing workshops and meetings, coordinating discussions, etc

Since this work is being carried out in the framework of the European Robotics *Research* Network, we have been more concerned with non-industrial scenarios. This report builds on previous work developed in EURON I [2] in which benchmarks in industry were discussed. In general this seems to be a different situation, since industry can provide the resources to measure whatever features they desire in a robot. In this sense they can not only develop their own benchmarks, but also they have even organized competitions: a famous example is the one held in March 1996, when Ford U.S.A. organized a competition for an order of 400 welding robots with the result that the KUKA robots could solve the benchmark problems considerably faster and smoother than the robots of the main competitor. KUKA won this contract and since then all Ford European plants become equipped with KUKA robots exclusively [3].

3 Actions

In order to attain the above-mentioned goals, namely successful robotics benchmarks in the medium term, the following actions were identified:

- a) Exhaustive survey: The first step is to make accessible to the community the state of the art in existing efforts in comparative research such as related initiatives in the U.S. and Japan, competitions, benchmarks, challenges, and conferences with relevant topics. Thus, information that was previously unavailable, or scattered in different sources and merged with irrelevant issues is make accessible to the community including a detailed description of the rules, metrics and procedures.
- b) Increase awareness by organizing and participating in workshops, meetings, and discussions; encourage experts in the different subfields to get involved in the process of benchmark definition, and follow these efforts suggesting further actions.
- c) Promote a set of benchmarking initiatives in some areas. Initially these sub-areas are selected by taking advantage of existing Special Interest Groups or technical committees, or when experts propose benchmarking initiatives. These benchmarks should be operational and described in sufficient detail, as well as their associated metrics, and an independent measurement procedure, keeping always in mind that the fundamental goal is that they are widely accepted in the community at large.

More concretely, we planned a series of discussions and refinements in parallel actions, similar to the procedure described in [1] as a continuous process of convergence towards consensus in order to ensure community wide acceptance:

- Identification of benchmarking initiatives in some areas
- Establish working groups (WG) one per area
- Discussion: on-line and physical
- First EURON Workshop on benchmarks in robotics research (Palermo, March 2006)
- Dissemination of on-going initiatives through a report and the web site.
- Discussion phase: online and physical open to the community at large
- Second Workshop on benchmarks in conjunction with an international event

The up-to-date results of action (a) above are described in the form of an exhaustive, detailed survey and inventory of current existing efforts in comparative research: competitions, benchmarks, challenges, repositories, conferences, etc. This has been the result of a long process of information gathering, either obtained from different sources or kindly provided by a number of persons. Most of this information was previously unavailable, scattered in different sources and merged with irrelevant issues. After a process of selection and rewriting it is made available to the community in a web site, located at:

<http://www.robot.uji.es/benchmarks>

Technical material available in this website may have interest for future benchmark development, since they include detailed description of the rules, metrics and procedures for current robot competitions and benchmarks.

Also actions (b) and (c) are being addressed with a number of meetings, workshops, and discussions — physical or email-based. The main overall result so far is a considerable increase in the awareness of the importance of robotics benchmarking in Europe. This has resulted in a number of on-going initiatives in Europe towards defining benchmarking scenarios and evaluation data sets, the current results of which are included in the web site. The degree of accomplished tasks varies among the different initiatives, some have been already available to the community for months on end, whereas others are more embryonic. Some of them are based on simulations only, and data sets are made available defining objects, robots and scenarios in standard format descriptions. When moving ahead beyond simulations into real hardware in the real world, computer data sets are not enough and various solutions are put forward. Some of them are based on specific hardware that is shared by remote access, whereas others describe experimental protocols to be shared in the verification of diverse approaches to the same problem.

Acknowledgements

This paper describes activities carried out at the Robotic Intelligence Laboratory of Universitat Jaume-I. Support for this work is provided by the European Commission through EURON the European Robotics Research Network.

References

[1] Burke, J.L., Murphy, R.R., Rogers, E., Lumelsky V.J., Scholtz, J., "Final Report for the DARPA/NSF Interdisciplinary Study on Human-Robot Interaction",

IEEE Trans. on Systems, Man, and Cybernetics Part C, 34, 103-112, 2004

[2] Dillmann, R. "Benchmarks for Robotics Research, EURON, April 2004. <http://www.cas.kth.se/euron/euron-deliverables/ka1-10-benchmarking.pdf>

[3] Ford Competition, <http://www.op.dlr.de/FF-DR-ER/research/irobot/optmp.html>

Robot Competitions — Ideal Benchmarks for Robotics Research

Sven Behnke

Humanoid Robots Group, Computer Science Institute
University of Freiburg, Georges-Köhler-Allee 52, 79110 Freiburg, Germany
e-mail: behnke@informatik.uni-freiburg.de

Abstract. In this paper, I argue for the use of robotic competitions as benchmarks for robotics research. By providing a common task to be solved at a specific place and a specific time, competitions avoid some of the difficulties arising when evaluating robotics research in the own lab. Competitions also bring together multiple research groups working on the same problem. This fosters the exchange of ideas. I review two of the most popular robotics competitions, RoboCup and the DARPA Grand Challenge, and discuss some issues arising when designing robotics competitions.

1. Introduction

Benchmarking robotics research is inherently difficult. Typically, results are reported only for a specific robotic system and a self-chosen set of tasks. The tasks are solved in the lab where the robot was developed. This makes it impossible to compare the results with other systems, developed in different labs and tested for different specific tasks. The commonly used "proof by video" technique has the same difficulties as the "proof by example" in other settings. That a robotic system works once in a video does not mean that it works always or that it works under slightly less controlled conditions.

One possible approach to overcome these shortcomings is to participate at robot competitions. Robot competitions bring together researchers, students, and enthusiasts in the pursuit of a technological challenge. Popular competitions include MicroMouse [1], where wheeled robots have to solve a maze, Robolympics [2], where robots compete in many disciplines, Robo-one [3], where remotely controlled humanoid robots engage in martial arts, and the AAAI Robot Competition [4], where robots have to solve different tasks in a conference environment. Among the most popular robot competitions are robotic soccer championships, like RoboCup [5] and FIRA [6], and competitions for unmanned ground and aerial vehicles, like the DARPA Grand Challenge [7], the European Land-Robot Trial (ELROB) [8], and the International Aerial Robotics Competition (IARC) [9]. Pobil compiled a survey of such competitions and other benchmarks for robotics [10]. Rainwater maintains a list of robot competitions [11].

Such robot competitions provide a standardized test bed for different robotic systems. All participating teams are forced to operate their robots outside their lab in a different environment at a scheduled time. This makes it possible to directly compare the different approaches for robot construction and control. In the following, I will review two of the most popular robotic competitions: RoboCup and the DARPA Grand Challenge.

2. RoboCup

RoboCup is an international joint project to promote AI, robotics, and related fields. The RoboCup Federation organizes since 1997 international robotic soccer competitions. The long-term goal of the RoboCup Federation is to develop by the year 2050 a team of humanoid soccer robots that wins against the FIFA world champion [12]. The soccer game was selected for the competitions, because, as opposed to chess, multiple players of one team must cooperate in a dynamic environment. Sensory signals must be interpreted in real-time and must be transformed into appropriate actions. The soccer competitions do not test isolated components, but two systems compete with each other. The number of goals scored is an objective performance measure that allows comparing systems that implement a large variety of approaches to perception, beha-

avior control, and robot construction. The presence of opponent teams, which continuously improve their system, makes the problem harder every year.

The RoboCup championships grew continuously over the years. In the last RoboCup, which took place in June 2006 in Bremen, Germany, 440 teams from 36 countries competed. The total number of participants was more than 2.600. In addition to the soccer competitions, since 2001, competitions for the search of victims of natural disasters and the coordination of rescue forces are held (RoboCupRescue). In 2006, for the first time, competitions for robots in a living-room environment took place in the RoboCup@home league. Furthermore, there are competitions for young researchers (RoboCupJunior).

The soccer competitions at RoboCup are held in five leagues. Since the beginning, there is a league for simulated agents, a league for small wheeled robots which are observed by cameras above the field (SmallSize), and a league for larger wheeled robots where external sensors are not permitted (MiddleSize). A league for the Sony Aibo dogs was added in 1999 (Four-legged) and a league for humanoid robots was established in 2002.

Different research issues are addressed in the different leagues. In the simulation league, team play and learning are most advanced. In the wheeled robot leagues, the robot construction (omnidirectional drives, ball manipulation devices), the perception of the situation on the field (omnidirectional vision systems, distance sensors), and the implementation of basic soccer skills (approaching, controlling, dribbling, and passing the ball) are still in the center of the activities. Because the robot hardware is fixed in the Four-legged League, the participating teams focus on perception and behavior control. Here, the control of the 18 degrees of freedom (DOF) poses considerable challenges.

As the performance of the robots increases, the competition rules are made more demanding by decreasing the deviations from the FIFA laws. This permanently increases the complexity of the problem. It can also be observed that solutions like team play, which have been developed in leagues abstracting from real-world problems, are adopted in hardware leagues, as the basic problems of robot construction, perception, locomotion, and ball manipulation are solved better.



Fig. 1. Some of the robots, which participated in the RoboCup 2006 Humanoid League competitions.

The Humanoid League is the most challenging RoboCupSoccer league. Its competition rules [13] require robots to have a human-like body plan. They consist of a trunk, two legs, two arms, and a head. The only allowed modes of locomotion are bipedal walking and running. The robots must be fully autonomous. No external power, computing power or remote control is allowed. After less demanding competitions, like walking around a pole and penalty kicks, at RoboCup 2005,

the first 2 vs. 2 soccer games were played in the KidSize class (30-60cm robot height). Fig. 1 shows some of the robots which participated in the RoboCup 2006 Humanoid League competitions.

Very different approaches for robot construction, perception, and behavior control were used. While some teams constructed their robots starting from commercially available kits, like Robotis Bioloid or Kondo KHR-1, many robots were designed from scratch by the teams. The largest and most expensive robot Arabot (Pal Technology, 30DOF, 140cm, 36kg) won the Footrace in the TeenSize class (65-130cm). Team NimbRo (Freiburg, Germany) constructed 20DOF, 60cm, 2.9kg robots, which won the KidSize Penalty Kick and came in second in the overall Best Humanoid ranking, the same result as in 2005. Winner of the Technical Challenge, the 2 vs. 2 soccer games, and the TeenSize Penalty Kick was Team Osaka, which used self-constructed robots with omnidirectional vision systems. Team Osaka was Best Humanoid for the third time in a row. This result shows that despite the variance caused by the randomness of soccer games, the RoboCup competitions do provide an objective performance measure.

3. DARPA Grand Challenge

The DARPA Grand Challenge benchmarks performance of autonomous ground vehicles. It is organized by the U.S. government to foster research and development in the area of autonomous driving. The first two competitions took place in the Mojave Desert. The course included gravel roads, paths, switchbacks, open desert areas and dry lakebeds, mountain passes, and some paved roadways. The course was outlined by a GPS corridor, which consisted of several thousand waypoints, accompanied by allowable path width and speed limits.

While following the GPS corridor, the vehicles had to recognize drivable surfaces by themselves and to make steering decisions in order to stay on the road and to avoid obstacles. Possible obstacles included other vehicles, fences, utility poles, stones, trees and bushes, and ditches. As skilled drivers with standard SUVs would have no difficulties driving the course, the challenge was information processing. Robustly perceiving the state of the environment and the vehicle, making driving decisions appropriate to the situation, and acting timely were key factors for success [14].

The vehicles had to be completely autonomous: no remote control capabilities were allowed. They could carry any combination of onboard sensors, both for sensing the position of the vehicle and the surrounding environment. Teams could also use any available, non-classified map and terrain database. The only external signals allowed were the pause and emergency stop remote control signals for the organizers and publicly available navigation aids, such as GPS signals and commercial differential correction services available to all teams.

The Grand Challenge events were divided into two segments: the qualification and the race. For qualification, the teams had to demonstrate the safety and reliability of their vehicles, including the emergency stop systems. They had to show autonomous motion capabilities on a test course which included narrow passages, obstacles, and a tunnel.

The first DARPA Grand Challenge took place on March 13th, 2004, but none of the participating vehicles came very far. On October 8th, 2005, 23 finalists started the second race, which was 213km long. This time, the participants were better prepared. The teams of Stanford University and Carnegie Mellon University (CMU), for example, drove prior to the race hundreds of km through similar terrain to calibrate and test their systems. Consequently, five autonomous vehicles finished the 2005 course. The \$2 million prize went to the fastest of them: Stanley [15] of Stanford Racing Team, which is shown in Fig. 2. It drove at an average speed of 30.7 km/h, with a top speed of 61km/h. The second and third fastest vehicles belonged to the Red Team of CMU.



Fig. 2. Stanley of Stanford Racing Team, winner of the 2006 DARPA Grand Challenge.

Major components of Stanley's software were based on machine learning, probabilistic reasoning, and real-time control. Probabilistic methods were necessary for robust perception in the presence of substantial measurement noise in the various sensors. Machine learning was applied prior to the race to tune system parameters and during the race to adapt to the terrain. Two important technical solutions developed for Stanley were adaptive road extraction from live camera images and speed adaptation. The idea for road extraction was that the road outside the range of the vehicles laser-scanners is likely to look similar to the surface classified as drivable in the laser range. This allowed planning the path further ahead than would have been possible with the laser range information alone. Speed adaptation was based on the supplied speed limits and imitation of human driver reactions to road conditions, like roughness, slope, and curvature.

While the vehicles in the 2005 challenge had to avoid static obstacles, they did not encounter moving obstacles, like other cars. The third competition, the DARPA Urban Challenge, is scheduled for November 3, 2007. It will take place in an urban environment with other traffic. The autonomous vehicles must not only find their way through a city, but they also must obey traffic laws.

4. Discussion

Robot competitions, as described above, proved to be a driving force of technological development. They allow for direct comparison of different approaches to solving a task. Participating teams are forced to leave their lab and to operate their robots at the competition site at the scheduled time. The competitive aspect unleashes huge energies and the competitions foster the exchange of ideas.

As the performance level of the robots rises, the competition rules must be developed to keep the challenges challenging. For competitions to be successful, it is important to ask for skills meaningful for many research groups. The skills to demonstrate must be challenging, but not too hard, as a hopeless challenge will not attract participants. Thus, there is a need for intensive exchange between organizers of the competition and the participants.

Naturally, robot competitions evaluate entire systems. When observing a difference of performance, it is frequently unclear, to which component of the systems the difference should be attributed to. A robot might not perform well for many reasons. It could be, for example, that the perception system is disturbed or that the behavior control software made a wrong decision or simply that an actuator is not working as designed. Hence, it is desirable to include in the competitions specific tests for subsystems.

Another issue is the availability of technical information about the winning systems. In order to advance the entire field, the teams should be required to release a detailed technical description after the competition. In some RoboCup leagues, where all teams share the same simulated or physical agents, it is even feasible to build on the software of the winning teams. To highlight technical advances, the competitions should be accompanied by technical conferences, where the underlying methods are discussed.

In conclusion, it can be stated that robot competitions are popular for good reasons. If designed well, they can be drivers for their field and ideal benchmarks for robotics research.

References

- [1] P. Harrison: Micromouse Information Centre. <http://micromouse.cannock.ac.uk>
- [2] ROBOlympics: <http://www.robolympics.net>
- [3] ROBO-ONE: Biped Robot Entertainment. <http://www.robbo-one.com>
- [4] S. Tejada and P. Rybski: AAAI 2005 Robot Competition and Exhibition. <http://palantir.swarthmore.edu/aaai05>
- [5] The RoboCup Federation: RoboCup Official Site. <http://www.robocup.org>
- [6] The Federation of International Robot-soccer Association (FIRA). <http://www.fira.net>
- [7] R. Kurjanowicz: DARPA Grand Challenge. <http://www.darpa.mil/grandchallenge>
- [8] Frank E. Schneider: European Land-Robot Trial ELROB. <http://www.elrob.org/>
- [9] Robert Michelson: AUVS International Aerial Robotics Competition. <http://avdil.gtri.gatech.edu/AUVS/IARCLaunchPoint.html>
- [10] Angel P. del Pobil: Research Benchmarks V1. EURON report, 2005.
- [11] R. Steven Rainwater: Robot Competition FAQ. <http://robots.net/rcfaq.html>
- [12] Hiroaki Kitano and Minoru Asada: The RoboCup Humanoid Challenge as the millennium challenge for advanced robotics. *Advanced Robotics*, 13(8):723-737, 2000.
- [13] S. Behnke: RoboCupSoccer Humanoid League Rules and Setup for the 2006 competition in Bremen, Germany. <http://www.humanoidsoccer.org/rules.html>
- [14] M. Montemerlo, S. Thrun, H. Dahlkamp, D. Stavens, and S. Strohband: Winning the DARPA Grand Challenge with an AI robot. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Boston, MA, 2006.
- [15] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, Nefian, and P. Mahoney: Stanley, the robot that won the DARPA Grand Challenge. To appear in *Journal of field Robotics*, 2006.

RAWSEEDS: Robotics Advancement through Web-publishing of Sensorial and Elaborated Extensive Data Sets

A. Bonarini*, W. Burgard†, G. Fontana*, M. Matteucci*, D. G. Sorrenti‡, J. D. Tardos§

* Department of Electronics and Information, Politecnico di Milano, I-20133 Milan, Italy
Email: bonarini,matteucci,fontana@elet.polimi.it

† Department of Computer Science, University of Freiburg, D-79110 Freiburg, Germany
Email: burgard@informatik.uni-freiburg.de

‡ Dip. Informatica Sistemistica e Comunicazione, Università degli Studi di Milano - Bicocca, I-20126, Milan, Italy
Email: sorrenti@disco.unimib.it

§ Depto. Informática e Ingeniería de Sistemas, Universidad de Zaragoza, S-50018 Zaragoza, Spain
Email: tardos@unizar.es

Abstract—The absence of standard benchmarks is an acknowledged problem in the field of robotics, and is doubly harmful to it. First, it prevents recognition of scientific and technical progress, thus discouraging research and development; second, it prevents new actors (and particularly companies) from entering the robotic sector, as heavy investments are needed to compensate for that absence. The need for benchmarking in advanced robotics embraces a wide range of topics, from, e.g., dexterous manipulation to, e.g., emotional interfaces with humans. The RAWSEEDS project will focus on sensor fusion, localization, mapping and SLAM in autonomous mobile robotics. The project will provide a comprehensive Benchmarking Toolkit, including high-quality multisensorial data sets, well defined Benchmarking Problems (BPs) based on the data sets, state-of-the-art Benchmarking Solutions (BSs) in the form of algorithms, software, methodologies and instruments for the assessment of the BSs.

I. INTRODUCTION

Progress in the field of robotics requires that robots and robotic systems gain the ability to operate with less and less direct human control, without detriment to their performance and, most importantly, to the safety of the people interacting with them. We are convinced that when robots will be able to safely navigate through environments designed for human beings, and to effectively execute tasks in those environments, beside and in collaboration with people, we will witness the birth of a new phase in the industrial development of the world. Just as the gradual transformation of computers from laboratory equipment to everyday, ubiquitous appliances created a gigantic market for Information Technology, when robots will lose their status of costly, complex, unsafe and “dumb” mechanisms, commercial applications for robotics will face an explosive growth. This, in turn, will lead to an enormous surge of interest - and financial resources - towards scientific and applied research in the whole field of robotics.

A key factor for a rapid progress towards this “robotic spread” is a substantial advancement in the performances of

robots associated to the concept of autonomy, i.e., the set of abilities needed to perform the following activities without human intervention:

- 1) perceive information about the environment;
- 2) extract from that information the elements needed to execute an assigned task;
- 3) decide which actions are needed to proceed towards the goal of the task;
- 4) correctly execute the selected actions;
- 5) manage the gap between the expected effect of the actions on the environment and their real effect, also taking into account the modifications of the environment not associated to the actions of the robot.

Among these, we consider that moving through its environment safely and without collision as well as being able to reach a goal location, is the basic ability that a robot must necessarily possess to autonomously operate. This requires, in particular, that the robot is capable to localize itself in the environment: this is usually done by constructing some form of internal representation of the environment, i.e., a map, and locating the position of the robot and its goal on the map. Any mobile autonomous robot must have the abilities needed to perform activities of mapping and self-localization, or even SLAM (Simultaneous Localization And Mapping), a well known problem in literature [7][8][9]. These abilities are not sufficient to ensure that the robot is also able to execute a task, but they can be thought of as necessary conditions for a mobile robot to be capable of effective autonomous behavior.

Solving the problems of mapping and self-localization is, unfortunately, not an easy task. One of the main problems is the fact that data elaborated by the robot come from sensors affected by imperfections, such as:

- limited spatial range and/or field-of-perception;
- noise;
- sensibility to spurious effects;

- low dynamic range;
- systematic errors or drift effects;
- failures.

These imperfections are very significant for any sensor, even costly state-of-the-art ones, but they become increasingly stringent as the cost of the sensors decrease, thus remarking the need for sophisticated sensor systems in advanced robotics. In the SLAM problem, for example, it is nowadays usual to use Laser Range Finders (LRF) as the main sensing system, also to recover 3D data from the environment (using a 3D LRF [10] system); however, in many mobile robots applications (outdoor navigation, indoor navigation with ramps in the environments, indoor cleaning, indoor navigation with obstacles not perceivable at the LRF height, door opening with door handle, semantic classification of places [11], etc.) an LRF cannot be the main robot sensing system or could integrate other perceiving devices. In many applications a vision-based sensing system (e.g. a trinocular vision system [13] or a correlation-based approach [14]), which could leverage on the huge amount of computer vision algorithms, can be the only complex sensing system of a mobile robot. This is also pushed by economical constraints: *“extensive market analyses show that a complex sensing system for a mobile robot cannot cost more than 10US\$, for a consumer-level robot”* [1].

Very sophisticated algorithms are then needed to process sensor output, elaborate it and extract the information needed to solve the mapping and localization problems. These algorithms become much more complex when multiple sensors are used (as is usually done to partially compensate for the intrinsic limitations of each sensor), because they need to include a process of sensor fusion between data coming from different sensors [5] [6]. Sensor fusion is mostly difficult when different kinds of sensors are employed (e.g. cameras and sonars), which is exactly what is generally done to explore different aspects of the environment and to exploit the capabilities of different sensor technologies [12]. Cheap sensors (such as the ones that present and future mass-market robotic applications are forced to employ for cost reasons) have very low performance and so, paradoxically, need the most sophisticated algorithms, as the data they generate must be subject to complex elaboration and interpretation procedures.

The ability to use cheap sensors and nonetheless build high-performance robotic products is absolutely necessary for the diffusion of mass-market robotic applications. However, the use of sophisticated algorithms does not necessarily have a significant impact on the final cost of a robotic product, as the main economic and conceptual effort is required for the development and test phases of the algorithms, while the implementation can be usually rely on inexpensive hardware. As we will show in the following section, presently the tools needed to design and develop such algorithms are not available to the vast majority of the (actually or potentially) interested groups: the objective of RAWSEEDS is to overcome this obstacle by realizing and making freely available such tools.

RAWSEEDS is a project funded by the European Commission as part of the VI EU Framework Program. It will have its official start on November 1st, 2006, and it will end on April 30th, 2009. The authors of this paper are the project's

proposers.

II. STATE OF THE ART

The study, design, engineering and marketing of autonomous robotic systems and solutions relies on the fact that the actors involved (mainly, research groups and companies) possess or can easily acquire the tools to develop and test sophisticated localization, mapping or SLAM algorithms. Such tools can be subdivided into the following categories:

- sensor data sets for the testing of systems on real-world environmental data;
- benchmarks and methodologies for the quantitative evaluation and comparison of algorithms performance;
- proven algorithms, having already demonstrated successful performances, to be used as starting points to develop new solutions and for comparison.

To be fully and readily useful, these elements would need to be integrated into a coherent Benchmarking Toolkit. This in turn requires: common and well-documented interfaces, immediate interoperability, extensive documentation, and accompanying support services.

Presently neither a toolkit of the kind described above nor its constituents are available to the general potential users. Some groups (essentially universities) have made the conspicuous investments needed to create for themselves some of the elements and shared their results with the community [2]. Even in these cases, the produced tools possess very limited performance and/or versatility and the usefulness of the crafted tools is strongly limited by the fact that they do not compare their results with those of others, because these have been obtained using different data sets or methodologies.

In fields not so distant from the topics of interest for roboticists, like computer vision, it is a little bit more common to refer to benchmarking toolkits, usually available on the web, as the reference source of data for the experimental sections of papers describing innovative scientific proposals. Example of such websites are those related to performance evaluation in traffic systems [3] and to 3D reconstruction [4].

Prospective actors (and especially companies), interested in entering the robotics field, find a strong deterrent in the difficulty and cost of acquiring the tools for the algorithm design described above. Presently, the only practical means to access to them (and to the know-how needed to develop new applications) is to set up a heavily funded (and years long) research program, without any possibility to evaluate in advance its eventual economical revenue. If a toolkit for the design, test and evaluation of sophisticated mapping, localization and/or SLAM algorithms would be made available at a low cost, this would in turn facilitate and speed up the activities of present operators in the robotic field, and encourage potential actors to join the drive towards new high-technology robotic research and products. This would then help to create the critical mass of knowledgeable operators and successful applications that will lead, in a short time, to the “robotic spread” that we have prefigured at the start of this Section.

RAWSEEDS will participate to this process by creating such a Benchmarking Toolkit, making it available for free and

disseminating it. Moreover RAWSEEDS (and particularly its website, described in the following sections) will propose itself as an instrument for the exchange of scientific results through the whole robotics community.

III. THE RAWSEEDS BENCHMARKING TOOLKIT

Advancement in any scientific and technical discipline relies on two basic mechanisms: competition between groups and exchange and dissemination of results among the research community. Both require that the results obtained by one group can be quantitatively evaluated by the other, and that the results obtained by the groups can be compared in order to find the best solutions. In the context of RAWSEEDS, the problem is essentially that of evaluating and comparing algorithms, which requires: (i) that the algorithms are applied to the same data and (ii) that an evaluation methodology exists. As we already discussed, even the first of these two conditions is presently very rarely fulfilled, but RAWSEEDS will offer a solution to this problem by providing comprehensive and validated multisensorial data sets. So the second condition, i.e. the availability of tools for the quantitative evaluation of algorithms, must be tackled: and this is what RAWSEEDS will do by creating suitable benchmarks.

In the context of the development of algorithms and software, a benchmark can be usually defined as a standard problem to which any algorithm in the considered class can be applied, together with a set of rules to evaluate the output produced. RAWSEEDS will generate and publish the data sets needed to define problems, and also two categories of structured benchmarks: Benchmark Problems (BPs) and Benchmark Solutions (BSs).

- A Benchmark Problem (BP) is defined as the union of: (i) a detailed and unambiguous description of a task; (ii) an extensive, detailed and validated collection of multisensorial data, gathered through experimental activity, to be used as the input for the execution of the task; (iii) a rating methodology for the evaluation of the results of the task execution. The application of the given methodology to the output of an algorithm or piece of software designed to solve a Benchmark Problem produces a set of scores that can be used to assess the performance of the algorithm or compare it with other algorithms.
- A Benchmark Solution (BS) is defined as the union of: (i) a BP; (ii) the detailed description of an algorithm for the solution of the BP (possibly including the source code of its implementation and/or executable code); (iii) the complete output of the algorithm applied to the BP; (iv) the set of scores of this output, obtained with the methodology specified in the BP.

The complete set of BPs and BSs published by RAWSEEDS is what we made reference to, in this document, as “RAWSEEDS Benchmarking Toolkit”. For instance: a Benchmark Problem may be a precise description of the task of extracting a map of an environment composed of line segments from the point-based representation of the environment produced by a laser range scanner, plus the complete scanner

data recorded on location, plus the rating methodology to be applied to the results. The union of this BP with an algorithm solving the problem (and possibly a software implementation of it), its results and their rating (obtained with by the given methodology) may then be a BS.

The main use of a BP is to clearly test existing (or in the course of development) algorithms. On the other hand, a BS can be very useful in many ways, as it will be possible to:

- compare the rating of the results obtained by the algorithm included in the BS with the rating obtained by another algorithm applied to the same BP (please remember that the rating methodology is defined by the BP itself, and so can be applied to different BSs);
- use the output of the algorithm included in the BS to get pre-processed input data for higher level algorithms to be tested, such as planners;
- use the algorithm included in the BS as a “building block” to design a complete multi-layer system for the processing of sensor data;
- use the algorithm included in the BS (and, if available, the source code of its implementation) as a source for the design of new, more sophisticated algorithms.

It must be noted that different BSs can be constructed for a single BP, so the number of BPs is not a limiting factor for the number of BSs that can be defined. Additionally, it is important to stress that the ratings of all the BSs based on the same BP, including the ones uploaded by users of the RAWSEEDS website, will be directly comparable. The BSs defined as part of the RAWSEEDS Benchmarking Toolkit will use state-of-the-art, well-proven algorithms that will constitute a corpus of “standard solutions” for the BPs and for similar problems. As we will see in Section 5, we foresee a conspicuous contribution of new BSs (and possibly new BPs) from the users of the RAWSEEDS website.

The sensory data (including vision data) contained in the BPs will be raw, i.e. they will not be subject to the application of any preliminary elaboration and/or compression procedure, to avoid *a priori* choices about what has to be considered “superfluous” or “redundant”. Data will be the result of exploration of a few representative environments by mobile test robots, capable of indoor and outdoor activity and fitted with an extensive set of sensor equipment, both of high and low quality, in order to accommodate the needs of both consumer and high-grade developments. This equipment will be chosen both to cover most of the common sensor schemata and quality levels used in autonomous robotics and also to provide the community with the most useful sets of high and low resolution raw data, that may be successively elaborated to extract higher-level environmental features and descriptions. RAWSEEDS plans to mount on the test robots the following sensor systems: 2 x 180 laser range scanners, trinocular and stereoscopic B/W camera systems, omnidirectional catadioptric camera, color cameras, sonars and GPS, in addition to suitable proprioceptive sensors (e.g. odometric systems, gyroscopes, accelerometers). Data acquisition will be performed by on-board PCs fitted with suitable data acquisition cards. If the necessity emerges, other sensors will be added to this list

during the project.

For the construction of the BPs, typical instances of different indoor and outdoor environments will be used, in both static (i.e., excluding moving elements such as people) and dynamic conditions. Each sensor data set will be collected moving the test robot through the environment on a complex exploratory path. The path will be covered with a succession of elementary “steps”: at the end of each step the raw output of each sensor or set of sensors (including cameras) will be recorded, and all the motions performed by the robot through the step (including accidental ones) will be logged. Each environment will be covered by multiple data sets, generated by performing exploration sessions on different paths with the same test robot; in this way it will be possible to use multiple data sets associated to the same environment to simulate a multi-robot data set. In our view the inclusion of outdoor locations is particularly significant, since many research groups do not own robot platforms capable to navigate through unstructured terrain and thus research results in this field are very scarce, even if many possible scientific and commercial applications can be envisaged.

Simply collecting high-resolution sensorial data is not sufficient to guarantee their precision and consistency, i.e. the fact that the data obtained from different sensor devices are coherent with each other, with the (logged) actions performed by the robot and with the physical environment explored. Moreover, advanced robotics applications require time coherence between different sensor data streams, which usually is neither guaranteed nor verifiable. To overcome these problems RAWSEEDS will include an extensive data validation phase, with the aim of verifying and certifying the consistency of the data produced by each sensor and their coherence with the ground truth. Statistical analysis of the data against the ground truth will be performed during the validation process, and its results (e.g. noise levels and distribution) added to the BPs.

IV. WWW.RAWSEEDS.ORG

A key part of the RAWSEEDS project is the construction of a website (URL: <http://www.rawseeds.org>). Users of the website (i.e. typically researchers and enterprises) will be able to download the BPs and BSs, as well as all their accessory information, such as: extensive description of all the hardware used to collect the data sets and of its configuration, ground truth, specifications of the data formats used, pictures of the test locations, notes. The site will include an extensive FAQ section, tutorials and instructions, and all the official documents of the RAWSEEDS project. It will also host a moderated forum, where researchers could exchange opinions and results as well as set up collaborations, and a “user section” where it will be possible to upload new BSs associated to the original BPs, or simply the set of results obtained by a novel algorithm applied to one of the BPs (evaluated with the methodology defined by the BP). The first option (upload a complete BS, including the description of the algorithm used to solve the chosen BP, and possibly the source code of the implementation used and/or executable files) will be probably preferred by

research groups, willing to share their results and see them acknowledged by the robotic community; the second option (publication of the results obtained, evaluated with the criteria defined by the BPs) is mainly dedicated to enterprises, which would prefer not to share how they can obtain such results. The possibility to upload new BPs will be left open, with the restriction that the environmental data included in the new BPs will have to certify their compliance to the same validation standards used (and documented) for the original RAWSEEDS BPs.

All that will be needed to become a user of the RAWSEEDS website is a free registration. During the registration process the user will be warned that any material submitted for publication will, if approved, become publicly available. The main goal of the RAWSEEDS website is to provide all the spectrum of data, from raw sensor output to the high-level representations obtained from that output by application of suitable (and immediately available as BSs) algorithms. These algorithms will be the ones provided by RAWSEEDS (a selection of the current state of the art) together with the new ones published by the users of the website. In this way, a current or prospective actor in the robotics field will have access not only to the data needed to immediately test its own applications (whichever be their abstraction level) without any need for a costly data-gathering programme, but also to alternative algorithms and results obtained by other groups. For example, a new application implementing symbolic models of environmental features could be tested simply by using as its input one of the maps produced by the published BSs.

As we will later show in more detail, RAWSEEDS will make efforts to promote among its users a general attitude towards the *sharing* of results; we believe, in fact, that it is the most useful for the progress of robotics and, in general, of science and technology.

A. Management of Intellectual Property Rights (IPR)

The upload section of the RAWSEEDS website will be designed to expand the range of possible users as widely as possible: for example, companies are usually very conservative towards the sharing of results with others, and thus need special attentions to be persuaded to do it. On the other side, we will restrict publication to those contributions that are really useful for the community, i.e. it will not be possible to use the site to “advertise” a result without sharing it, at least partially, with the other users. To reach this objective, a carefully tailored IPR policy will be adopted.

The material that will be published by the site can be divided into two broad categories:

- material produced by the RAWSEEDS project itself;
- material voluntarily submitted for publication by the users of the website.

Both will be subject to the same IPR regime, RAWSEEDS will require that any material published on the website complies with the following three requisites:

- R1: its creator chose to make that material publicly available;

- R2: the rights granted by the creator to the users of the material are clearly and explicitly defined;
- R3: the material must qualify as useful and appropriate for publication.

Requisite R1 is automatically satisfied by the fact that it is the free choice of each user of RAWSEEDS if he/she wants to publish any of his/her creations, and which ones.

To meet requisite R2, any contributor to www.rawseeds.org will have to accompany the proposed material with a *license* stating which rights he/she reserves to himself/herself and which are instead granted to the public. So intellectual property of any material published by RAWSEEDS will remain to its creator, who (with the act of submitting the material for publication) will choose to relinquish part of the rights on it to the public. Which rights are actually given to the public is defined by the chosen license. RAWSEEDS will leave the choice of the license to the user, with the only constraint that a copy of the chosen license is sent along with the material submitted for publication.

To meet requisite R3 any contribution will have to be previously examined and approved by the administrators of www.rawseeds.org, prior to publication. They will decide about the publication of each contribution, and their judgement will be based upon the following definition:

Any publishable material is considered *useful and appropriate* for publication on the RAWSEEDS website if all of the following are true:

- 1) it is related and can help progress in the field of robotics, especially regarding Localization, Mapping and SLAM;
- 2) it is sufficiently detailed to be usable (e.g., the description of an algorithm must be complete enough to allow a reader to implement - that algorithm into a piece of software);
- 3) it is usable (e.g., executable code is usable only if it is actually working and accompanied by all the information needed to install and configure it);
- 4) it does not have commercial purposes only (e.g., a company marketing robotic products could publish the description of a product, but that description will need to disclose enough data about the product to be considered a worthwhile contribution to the field in itself rather than a marketing operation).

B. Dissemination actions and feedback

As the results of RAWSEEDS have the objective of being useful tools for all the actors involved into the development of robotics, rather than scientific achievements *per se*, feedback from the robotics community is explicitly sought. In particular, the RAWSEEDS workplan includes an extensive preliminary phase aimed to the precise definition of the Benchmarking Toolkit that the project will later develop, in order to maximise the usefulness of the toolkit: therefore any motivated suggestion about this topic will be welcome, and (within the limits of the project as defined by the contract between the EU and the proposers) we will take due account of all such suggestions during the benchmark definition phase. Any suggestion can be already sent to suggestions@rawseeds.org.

To maximise the impact of RAWSEEDS, suitable dissemination activities have been expressly included in the workplan; they will be publicized as soon as they are organized. For the same reason RAWSEEDS is proposed by a consortium of partners with an acknowledged expertise in the research fields covered by the project, and coming from different countries.

V. CONCLUSION

RAWSEEDS is a project aimed at overcoming one of the main limits to research and development in robotics, i.e. the lack of comprehensive, validated and publicly available benchmarks. It will reach its objective through the development of a complete Benchmarking Toolkit and the setup of a website (<http://www.rawseeds.org>). The website will publish not only the RAWSEEDS Benchmarking Toolkit, but also any useful extension of it coming from the scientific and technical community, and all of this material will be freely available. To help the sharing of results and the general progress of the robotic field, the RAWSEEDS website will also act as a point of aggregation and exchange of information, that will be flanked by a suitable program of dissemination activities.

REFERENCES

- [1] Colin Angle, *Invited talk at EURON meeting*, Amsterdam, 2004.
- [2] Andrew Howard and Nicholas Roy, *The Robotics Data Set Repository (Radish)*, <http://radish.sourceforge.net/>, 2003.
- [3] Computational Vision Group, *PETS: Performance Evaluation of Tracking and Surveillance*, The University of Reading, <http://www.cvg.cs.rdg.ac.uk>.
- [4] D. Scharstein and R. Szeliski, *Stereo Vision Research Page* <http://www.middlebury.edu/stereo>, 2002.
- [5] Hugh F. Durrant-Whyte, *Integration, coordination and control of multi-sensor robot systems*, Kluwer Academic, 1987.
- [6] Juan D. Tardós and Jose A. Castellanos, *Mobile robot localization and map building : a multisensor fusion approach*, Kluwer Academic, 1999.
- [7] S. Thrun and M. Montemerlo and D. Koller and B. Wegbreit and J. Nieto and E. Nebot, *FastSLAM: An Efficient Solution to the Simultaneous Localization And Mapping Problem with Unknown Data Association*, Journal of Machine Learning Research, 2004.
- [8] J. Folkesson and Henrik I. Christensen, *Robust SLAM*, 5th IFAC Symp. on Intelligent Autonomous Vehicles, 2004.
- [9] F. Lu and E. Milios, *Globally consistent range scan alignment for environment mapping*, Autonomous Robots, 1997.
- [10] Oliver Wulf and Bernardo Wagner and Mohamed Khalaf-Allah, *Using 3D data for Monte Carlo localization in complex indoor environments*, Proc. of ECMR Conf., 2005.
- [11] Rottmann, A. and Martínez Mozos, O. and Stachniss, C. and Burgard, W., *Place Classification of Indoor Environments with Mobile Robots using Boosting*, aaai, Pittsburgh, PA, USA, 2005.
- [12] J. Castellanos and J. Montiel and J. Neira and J. Tardos, *The SPmap: A probabilistic framework for simultaneous localization and map building*, IEEE Transactions on Robotics and Automation, 15(5):948–953, 1999.
- [13] N. Ayache and F. Lustman, *Trinocular Stereo vision for Robotics*, IEEE Trans. on PAMI, Vol.12-1, 1991.
- [14] Kurt Konolige *Small vision system. hardware and implementation*, isrr, Hayama, Japan, 1997.

Steps Towards the Automatic Evaluation of Robot Obstacle Avoidance Algorithms

I. Rañó

*Dpto. de Informática e Ing. de Sistemas
Universidad de Zaragoza, Spain
irano@unizar.es*

J. Minguez

*I3A, Dpto. de Informática e Ing. de Sistemas
Universidad de Zaragoza, Spain
jminguez@unizar.es*

Abstract—This paper presents the first steps towards the evaluation of obstacle avoidance techniques for mobile robots. The idea is to create a methodology to evaluate the performance of the methods given a wide range of work conditions. The work conditions usually include scenarios with very different nature (dense, complex, cluttered, etc). The performance is measured in terms of robotic parameters (robustness, optimality, safety, etc). We describe in this paper the overall methodology that we intend to apply and the first steps in the scenario characterization.

I. INTRODUCTION

The *rationale* of the workshop explains very well the objective of this research:

Current practice of publishing research results in robotics makes it extremely difficult not only to compare results of different approaches, but also to assess the quality of the research presented by the authors. Though for pure theoretical articles this may not be the case, typically when researchers claim that their particular algorithm or system is capable of achieving some performance, those claims are intrinsically unverifiable, either because it is their unique system or just because a lack of experimental details, including working hypothesis. (...).

It is clear that to overcome this issue, we have to find ways or processes to automatically evaluate the research with methodologies accepted by the community. Research on this topic forks into a *top-down* or *bottom-up* perspectives. On the one hand, the top-down manner consists in evaluating the complete robot performance when developing a given task (see [5], [3] and the majority of the speakers in the *First European Workshop on Benchmarks in Robotics Research* in 2006 Euron meeting). The advantage of this strategy is that it results straightforward to assess whether the robot has accomplished a given task. However, the disadvantage is that this requires a strong experimental validation with the real robots. Thus, rigorous protocols of experimentation have to be developed to deal with the repeatability problem of real experiments and to guarantee the checking of all possible situations.

On the other hand, the bottom-up manner deals with the evaluation of each single subtask individually. The aim is to explain the robot performance developing a task as the synergy of each particular performance in each of the involved subtasks. This paper follows this direction and focusses on

the automatic evaluation of collision avoidance, which is a particular subtask involved in many applications of mobile robots.

This work is a part of a Spanish project. The objective is the evaluation of robots for mobility aid (like electric walkers or robotic wheelchairs). There are two main research axes: (i) the evaluation of robot motion from a robotic perspective and (ii) the evaluation of robot motion from a human-centered point of view. This paper describes the first steps towards the automatic evaluation of motion in the first axis, and the complete overview of the methodology that we will try to use.

II. ROBOT OBSTACLE AVOIDANCE AND EVALUATION PERSPECTIVE

This work focusses in one of the fundamental modules of sensor-based motion schemes: reactive obstacle avoidance. This module is the responsible of moving a vehicle to a given goal location while avoiding collisions with the scenario. Usually, it is the last responsible of the motion.

From a robotic perspective, there are many techniques that have been designed to address autonomous collision-free motion (sensor-based motion with obstacle avoidance). For example [4], [1], [2], [12], [8] among many others. It is clear that under the same conditions each technique generates a different motion. Nevertheless, questions like: which is the most robust one? or which of them behaves better in a determined context or condition? cannot be answered neither from a scientific nor technological point of view. In other words, once we face a mobile robotics application, the selection of a motion technique among all the existing ones is a matter of specialists and not accessible to everybody. This is because there are not objective comparisons of methods neither quantitative (in terms robustness or action parameters of such as the time or the total distance traveled) nor qualitative (in terms of security of the motion). At present, there is only one experimental comparison [6]. Nevertheless, this comparison is very old, and thus, it does not include the advances in this subject in the last 15 years. Furthermore it is based on the observation and does not present a rigorous and objective methodology to address this objective.

III. OVERVIEW OF THE EVALUATION FRAMEWORK

An obstacle avoidance technique is a mechanism that given an *obstacle configuration* and a *initial* and *goal* configurations computes the best motion to drive the vehicle to the goal while avoiding collisions with the obstacles. This process is repeated until the goal is reached. The result is a *trajectory* that joins the initial and the goal but generated online (since they work in an iterative process). Thus, the **inputs** of the problem are: (i) obstacle distribution and (ii) initial and goal configurations. The **output** is *success* if a collision free trajectory that joins the initial and goal has been computed. *Failure*: collision or the goal is not reached (usually referred as local minima). Thus, an evaluation of an obstacle avoidance mechanism should cover all the possible inputs (obstacles and initial and goal locations) and must be done on the basis of the quality of the output (collision free trajectory).

The methodology that we propose is to build a system able to generate random obstacle, initial and goal distributions and to evaluate the output of the method (trajectory) as a function of quantitative descriptors of the inputs and the outputs. The framework has the following modules (Figure 1):

- 1) *Scenario generation*: random generator of obstacle distributions and initial and goal locations.
- 2) *Scenario characterization*: extraction of the quantitative descriptors of the environment (e.g. density, clearness, etc).
- 3) *Collision Avoidance*: this is the technique to evaluate and is a “black-box” in the framework.
- 4) *Robot Simulator*: simulates the next state of the robot given the motion computed by the collision avoidance technique. To simplify the problem, in this work we assume that the robot is circular (with radius R) and holonomic without dynamic constraints. The sensor is assumed to measure range.
- 5) *Trajectory Evaluation*: extraction of the quantitative descriptors of the trajectory (e.g. optimality, safety, etc).

The final step is to describe the performance of the method as a function of this quantitative parameters. Notice that the evaluation measures how a given technique behaves for different environmental conditions. The environmental conditions are expressed in terms of density, confinement, clearness, etc. The behavior is described in terms of the success ratio, safety and optimality, etc.

The validation is performed exploring as much as possible the variability of the scenario. This forces the proposed validation methodology to work only on simulation, since the cost of evaluation on real environments is prohibitive. Notice that this process extrapolates for different methods giving an adequate framework for comparison.

We next describe the modules in detail.

A. Scenario generator

The scenario generator module creates a random obstacle distribution and goal and initial locations. These entities are represented geometrically as follows.

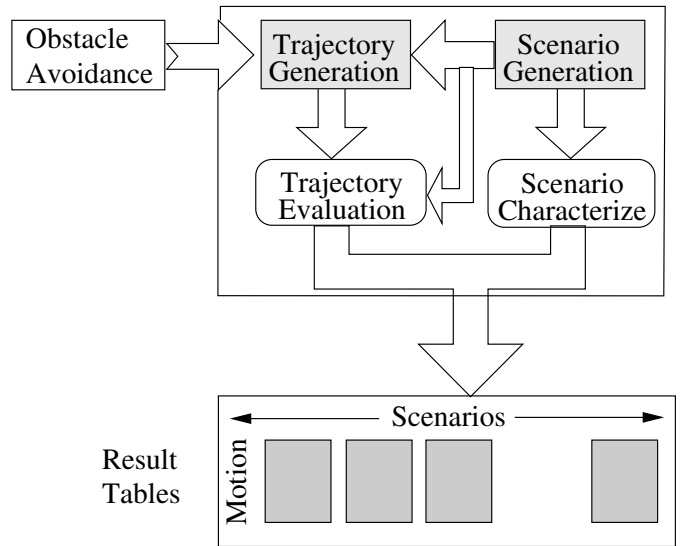


Fig. 1. This figure shows the modules of the evaluation framework

The scenario is a unitary ball with Euclidean metric. The obstacles are spheres within the unit ball. This is because spheres are a base of \mathbf{R}^2 and many shapes can be constructed with spheres (e.g. general polygons can be represented by circles [11]). In other words, the selection of circular obstacles is not a limitation. The number of obstacles is random. Furthermore, for each obstacle the location of the center and the radius are also random. Figure 2 shows two examples.

The initial and goal locations are also randomly generated within the unit ball. If any of the locations is within an obstacle, this location is recomputed. Furthermore, for any obstacle distribution and initial and goal locations, it is important to check the existence of a collision free path. Otherwise there is no solution for the obstacle avoidance technique. This is performed through a complete planner.

B. Scenario Characterization

This module extracts a quantitative evaluation of some descriptors of the environment. The aim of the scenario characterization is to extract numerical values that represent qualitative scenario features with a quantitative intrinsic values. The descriptors must be both related to the common

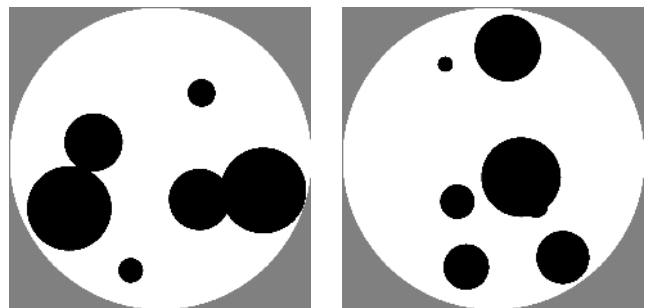


Fig. 2. Two examples of random scenarios with six and seven obstacles.

intuition of the qualitative variables that they define, and their values must be correlated with the performance of the obstacle avoidance algorithms. Table I describe the parameters and the human intuition qualitative descriptors.

Human Descriptor	Mathematical Parameter
Density	Density
Clearness	Dispersion
Confinement	Nearest Neighbor Metric
Uniformity	Discrepancy
Clutterness	Aleatory
Structure	Convex Hull
Others...	...

TABLE I
DESCRIPTORS OF THE SCENARIO.

We define these descriptors next. Some of them have been already developed while others need a deeper study.

a) Density: The density measures the amount of space occupied by obstacles. This is an intrinsic and global property of the environment, independent of the robot shape and size. Let be C_u the unitary sphere and lets assume a given distribution of spheres $\mathcal{C} = \{C_i\}$. The density of occupied space is:

$$\rho(\mathcal{C}) = \frac{A(\bigcup_i C_i)}{\pi} \quad (1)$$

where C_i is the i -th circle, and function $A(\cdot)$ computes the area of a set of circles. Notice that the density is $\rho \in [0, 1]$. In other words is normalized. On one hand, when $\mathcal{C} = \{\emptyset\}$ the area is zero, hence the density is also zero. On the other hand, if $\mathcal{C} = \{C_u\}$ the density of obstacles is one, which makes the scenario completely occupied and leaves no free space for the robot motion. Table II depicts some examples.

b) Clearness: The clearness is related to the maximum open space among the obstacle distribution. This descriptor depends on the size of the robot. One way of measuring open areas in distributions is the dispersion [9], since it measures the biggest obstacle free ball among the distribution. In other words, the value is the radius of the largest circle on the free space. Let be $\tilde{\mathcal{C}} = \mathcal{C} \oplus C_{\frac{R}{2}}$, where \oplus is the Minkovski sum of sets and $C_{\frac{R}{2}}$ is the sphere with radius $\frac{R}{2}$. This set is the obstacle distribution enlarged the radius of the robot. The dispersion of \mathcal{C} is:

$$\delta(\mathcal{C}) = \sup_{p \in C_u} \left\{ \min_{C_i \in \mathcal{C}} \{ \|p - \tilde{C}_i\| \} \right\} \quad (2)$$

where $\|\cdot\|$ is the Euclidean distance from point p to the sphere \tilde{C}_i ($\tilde{C}_i = C_i \oplus C_{\frac{R}{2}}$). Notice that the dispersion $\delta \in [0, 1]$. When there are no obstacles the value of the dispersion is one. However, as the number of obstacles and their radius increase the dispersion drops to zero. This characteristic captures the notion of clearness (open space) since it represents the maximum allowable distance for the robot to maneuver. Table II depicts some examples.

c) Confinement: The notion of confinement is related with the lack of space to manoeuvre (the distance between obstacles). This notion depends on the robot size reason why we use \tilde{C} . A measure is:

$$\kappa(\mathcal{C}) = 1 - \frac{1}{n} \sum_{i=1}^n \frac{d_{sph}(\tilde{C}_i - NN(\tilde{C}_i))}{2}, \quad n > 1 \quad (3)$$

where d_{sph} is the Euclidean distance between two spheres and $NN(C_i)$ is the closest sphere to C_i (nearest neighborhood). Notice that the confinement is $\kappa \in [0, 1]$. High values of confinement explain obstacles which are very close among them, while low values are due to far obstacles.

d) Uniformity: The uniformity in the obstacle distribution refers to the match with a uniformly distributed set of obstacles. In fact this is measured by the discrepancy. Let be C_r the set of balls with radius r in the unit circle C_u . The discrepancy is:

$$\eta(\mathcal{C}) = \sup_{r \in [0,1]} \left\{ \left| \frac{A(\mathcal{C} \cap C_r)}{A(\mathcal{C})} - \frac{A(C_r)}{\pi} \right| \right\} \quad (4)$$

Notice that the discrepancy $\eta \in [0, 1]$. When there are no obstacles both $A(\mathcal{C}) = 0$ and $A(\mathcal{C} \cap C_r) = 0$ (discrepancy is not defined). Low values of discrepancy represent well distributed obstacle points. The discrepancy tends to one as the obstacles are closed and not equally distributed distributed in the space. Table II depicts some examples.

e) Cluttering: The cluttering refers to the order of the distribution. The disorder on a obstacle distribution is related with the randomness. We plan to use information theory to measure the randomness of sequences by using the tests of Martin-Lof [7] or Kolmogorov complexity of constructive measurement. Other works use the entropy as a descriptor [5] to measure the disorder. However, from our point of view entropy is a local descriptor that needs to approximate a probability distribution, which seems difficult to obtain from an obstacle distribution.

f) Structure: The structure of the scenario measures the the tendency of the scenario to approximate a polygonal world (man-made scenarios). The measure is, for all clusters, the normalized difference between a cluster of obstacles C^j (set of connected spheres) and its convex hull. The measure of the structure is

$$\gamma(\mathcal{C}) = 1 - \frac{1}{N} \sum_{j=1}^N (A(\hat{C}^j) - A(C^j)) \quad (5)$$

where \hat{C}^j the convex-hull of C^j and N the number of clusters. Notice that $\gamma \in [0, 1]$. This measure gives an idea of the structure underlying the cluster. For example the value is one for a perfect line and zero for sparse non intersecting obstacles. We have notice that there are still some issues to fix with this measurement since it fails to represent well aligned non convex polygons. We are trying to correct it by sub-clustering each cluster with alignment criteria.

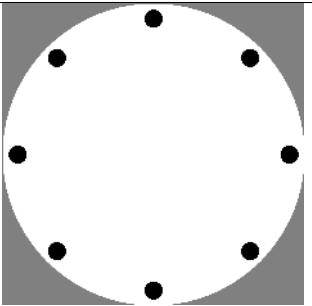
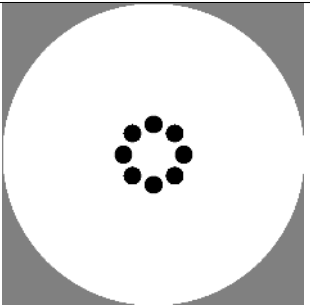
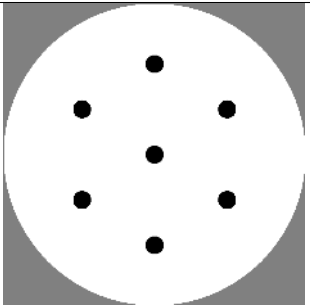
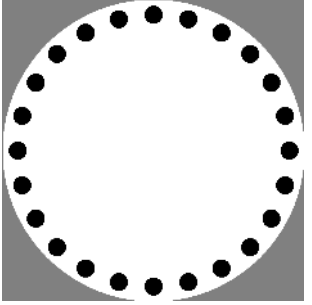
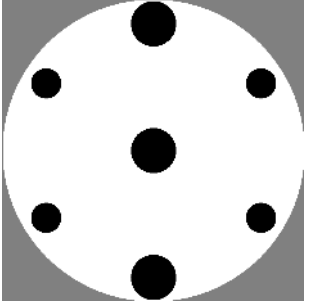
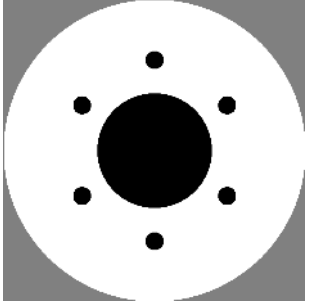
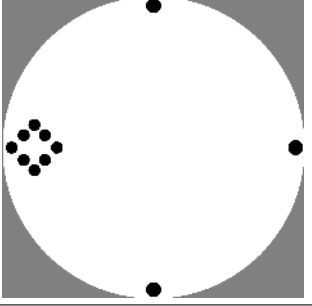
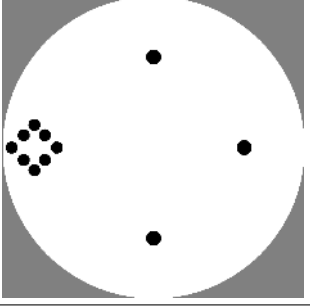
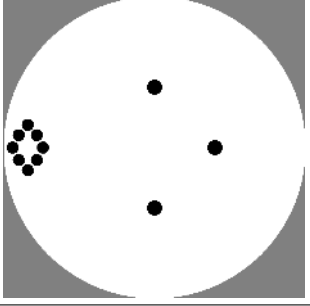
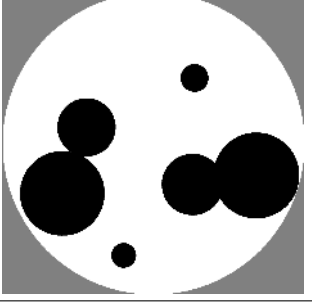
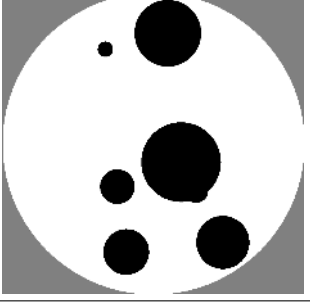
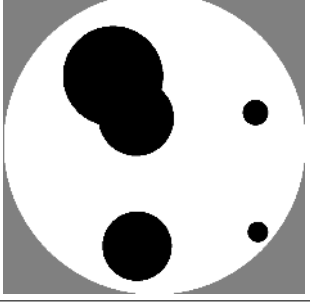
Environment			
Density	0.0287	0.0287	0.0252
Clearness	0.840	0.381	0.307
Confinement	?	?	?
Uniformity	0.138	0.447	0.184
Cluttering	?	?	?
Structure	?	?	?
Environment			
Density	0.086	0.107	0.166
Clearness	0.840	0.377	0.307
Confinement	?	?	?
Uniformity	0.137	0.139	0.244
Cluttering	?	?	?
Structure	?	?	?
Environment			
Density	0.020	0.020	0.020
Clearness	0.757	0.552	0.431
Confinement	?	?	?
Uniformity	0.390	0.390	0.425
Cluttering	?	?	?
Structure	?	?	?
Environment			
Density	0.251	0.189	0.210
Clearness	0.374	0.437	0.400
Confinement	?	?	?
Uniformity	0.461	0.379	0.444
Cluttering	?	?	?
Structure	?	?	?

TABLE II

CHARACTERIZATION OF SCENARIOS. THE SYMBOL ? MEANS THAT THE VALUE HAS NOT BEEN COMPUTED YET.

C. Robot Simulator

This module emulates the sensory and motion processes of the motion:

- 1) Sensory process: given the obstacle distribution and the current vehicle location, this module computes the *generalized visibility polygon* as generic range sensory measurement.
- 2) Motion process: given the motion computed by the obstacle avoidance method, this module computes the next robot state in a given period of time δt .

D. Trajectory Evaluation

This module extracts quantitative descriptors of quality of the trajectory. The aim of this characterisation is to extract numerical values that represent a qualitative measurement of the trajectory. We denote the trajectory generated by the method:

$$\begin{aligned} \chi : [0, 1] &\mapsto C_u - \mathcal{C} \\ \phi &\mapsto \chi(\phi) = \mathbf{q} \end{aligned} \quad (6)$$

where $\chi(0) = \mathbf{q}_{\text{init}}$ and $\chi(1) = \mathbf{q}_{\text{goal}}$. We describe next some of the parameters.

a) Success: This is the most important parameter since it describes the success of the task. Notice that the collision avoidance techniques are local techniques, and thus they could get trapped in local minima (not reaching the goal) or even to have collisions. In both cases ($\chi(1) \neq \mathbf{q}_{\text{goal}}$ or $\exists \phi$ such that $\chi(\phi) \in \mathcal{C}$) the result is *failure*. Then the success is $\eta(\mathcal{C}, \mathbf{q}_{\text{init}}, \mathbf{q}_{\text{goal}}) = [\{0\}, \{1\}]$. If the obstacle avoidance mechanism fails $\eta = 0$ otherwise $\eta = 1$. From now on the rest of the parameters are defined when $\eta = 1$.

b) Optimality: This parameters measures how the trajectory matches the optimal path. In order to compute the optimal trajectory χ_{opt} we have adapted the visibility graph technique [10] to work in spherical worlds¹. Let be Φ an *optimality* function defined over the space, such that at each \mathbf{q} , $\Phi(\mathbf{q})$ is the length of the path with minimum length that joins \mathbf{q} and the optimal trajectory χ (without lying in \mathcal{C}). Then there are some concepts that give the “difference” with the optimal path like the difference of *lengths* of the trajectories, or how the trajectory differs from the optimal by integrating $\Phi(\chi(\phi))$. We plan to implement both parameters.

Figure 3a,c show two different scenarios and the trajectories computed by an obstacle avoidance method (potential field method) and the optimal trajectory to the problem.

c) Safety: The safety measures how close the trajectory matches the safest trajectory. Notice that usually safest is far different from optimal (an optimal trajectory usually graze the obstacles, which is the un-safest trajectory). In order to compute the safest trajectory, we compute the Voronoi diagram of the obstacle distribution. Then, we define a Voronoi function

¹The path is a sequence of straight lines and circular arcs instead of straight lines

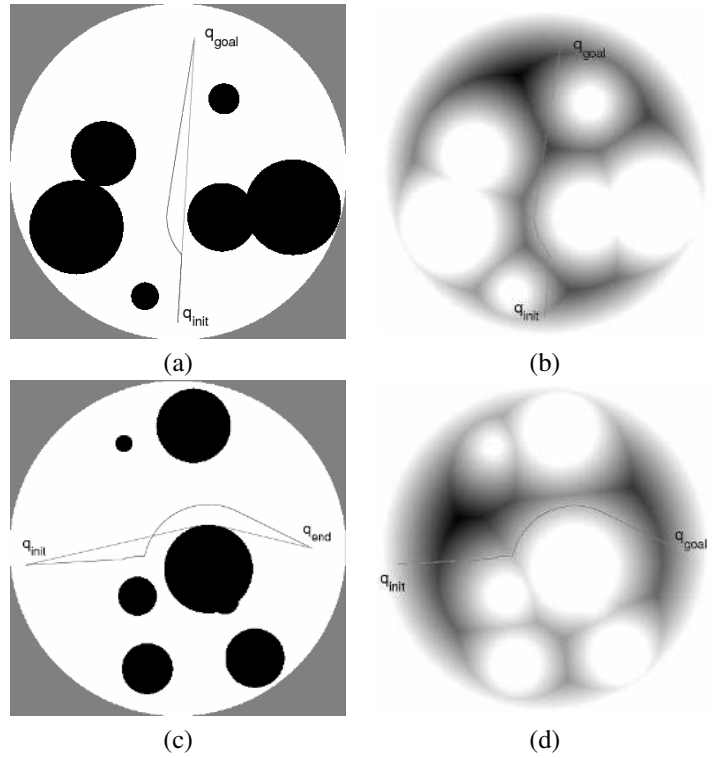


Fig. 3. This figures display the optimality function and the Voronoi function, and real trajectory generated with a PFM method on scenario from two scenarios.

$Vor(\mathbf{q})$ that takes values in all the space. The function is the length of the path with minimum length that joins \mathbf{q} and the Voronoi diagram (without lying in \mathcal{C}).

Then there are some concepts that give the “difference” with the safest path (the path on the Voronoi diagram) like the difference of *lengths* of the trajectories, or how the trajectory differs from the safest by integrating $Vor(\chi(\phi))$. We plan to implement both parameters.

Figure 3b,d show two different scenarios, the trajectory computed by an obstacle avoidance method (potential field method) and an approximation of the Vor function.

d) Other Characteristics: Other parameters could be defined to characterize the trajectories, but they need to be matched against a ground truth. In general a good avoidance algorithm must display a tradeoff over the above mentioned characteristics, since for instance safety and path length optimality are usually a tradeoff. The characteristics have been restricted to the cinematic domain, other parameters like time optimality of a path are clearly dependent on the dynamics of the robot, and cannot be directly expressed under the present assumptions.

While the scenarios considered here are static, another interesting characteristic to take into account for dynamic environments is the speed of the response to an unexpected change. The latency of the algorithm is indeed an important issue in dynamic environments, which is related to the amount of history information internally stored on the algorithm. A purely reactive algorithm will show a better latency equal to

Density (ρ) VS Success	[0, 0.25)	[0.25, 0.5)	[0.5, 0.75)	[0.75, 1]
Success	95%	80%	68%	41%
Failure	5%	20%	32%	59%

TABLE III

METHOD 1: DENSITY VERSUS SUCCESS RATE.

Density (ρ) VS Success	[0, 0.25)	[0.25, 0.5)	[0.5, 0.75)	[0.75, 1]
Success	95%	95%	60%	20%
Failure	5%	5%	40%	80%

TABLE IV

METHOD 2: DENSITY VERSUS SUCCESS RATE.

the computation time. This delay in the response is also related with a parameter that can be measured on the framework, even it has not been yet mentioned, the execution time of the avoidance cycle.

E. Final evaluation

The above mentioned process is repeated for a significant number of scenarios (obstacle distributions, initial and goal locations) for each of the methods X_i to evaluate. The final results should express the working conditions of each method. For each pair of descriptors (scenario / trajectory) the benchmark procedure produces a table.

We next describe some examples based on imaginary data. Table III and Table IV represent the scenario density versus the success ratio for methods X_1 and X_2 . These tables describe how the algorithms behave for a different range of scenario density (as usual, the robustness of the method decreases as the density increases). This is very useful for comparison and selection. For example, if in the application the range of density is low, Method 2 is more robust than Method 1. However, if the density is larger, the Method 1 become more robust. If the density of the scenario can be a priori estimated, the selection is clear from these tables (even for non experts).

Table V represents the normalized optimality parameter for different density ranges. This table describes how the optimality in length of the paths generated change as a function of the density of the scenario.

Notice that from this evaluation one can extract conclusions about a given method and compare the performance of the methods among them.

IV. CONCLUSIONS AND FURTHER WORK

This paper presents the first steps towards the evaluation of obstacle avoidance algorithms for mobile robots. We understand that the benefits of this evaluation are twofold. On the one hand, it is useful for researchers and developers to have rigorous evaluation tools as a objective feedback of their designs. On the other hand, for non technical experts, the decision of what method is well suited for a given application

Density VS Optimality Param.	[0, 0.25)	[0.25, 0.5)	[0.5, 0.75)	[0.75, 1]
[0, 0.25)	65%	50%	40%	25%
[0.25, 0.5)	20%	25%	10%	10%
[0.5, 75)	10%	15%	5%	5%
[0.75, 1]	5%	10%	3%	1%

TABLE V

METHOD 1: DENSITY VERSUS OPTIMALITY.

is a matter of understanding the working conditions of the application, but not the technical details of each technique.

The project that includes this work is currently starting, hence only first steps results are presented. Even the overall evaluation methodology has been depicted, only parts of the scenario characterization has been actually obtained. We are aware that more characteristics must be defined as this is an actually ongoing project. However the defined characteristics seem to properly catch intuitive concepts about environments, which is a key element of obstacle avoidance evaluation.

REFERENCES

- [1] J. Borenstein and Y. Koren. The Vector Field Histogram—Fast Obstacle Avoidance for Mobile Robots. *IEEE Transactions on Robotics and Automation*, 7:278–288, 1991.
- [2] D. Fox, W. Burgard, and S. Thrun. The Dynamic Window Approach to Collision Avoidance. *IEEE Robotics and Automation Magazine*, 4(1), 1997.
- [3] J. Held, A. Lampe, and R. Chatila. Linking mobile robot performances with the environment using system maps. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, 2006.
- [4] O. Khatib. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *Int. Journal of Robotics Research*, 5:90–98, 1986.
- [5] A. Lampe and R. Chatila. Performance measure for the evaluation of mobile robot autonomy. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2006.
- [6] A. Manz, R. Liscano, and D. Green. A comparison of realtime obstacle avoidance methods for mobile robots. In R. Chatila and G. Hirzinger, editors, *Experimental Robotics II*, pages 301–316. Springer-Verlag, 1991.
- [7] P. Martin-Lof. The definition of random sequences. *Information and Control*, 9:602–619, 1966.
- [8] J. Minguez and L. Montano. Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios. *IEEE Transactions on Robotics and Automation*, 20(1):45–59, 2004.
- [9] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, Philadelphia, 1992.
- [10] N.J. Nilsson. A mobile automaton: An application of artificial intelligence techniques. In *Proceedings of the 1st International Joint Conference on Artificial Intelligence*, pages 509–520, 1969.
- [11] J. O’Rourke and N. Badler. Decomposition of three-dimensional objects into spheres. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(3):295–305, 1979.
- [12] R. Simmons. The Curvature-Velocity Method for Local Obstacle Avoidance. In *IEEE Int. Conf. on Robotics and Automation*, pages 3375–3382, Minneapolis, USA, 1996.

On Experimental Research in Sampling-based Motion Planning

Roland Geraerts

*Institute of Information and Computing Sciences, Utrecht University
P.O. Box 80.089, 3508 TB Utrecht, the Netherlands, Email: roland@cs.uu.nl*

1 Introduction

Motion planning is one of the fundamental problems in robotics. The motion planning problem can be defined as finding a path between a start and goal placement of a robot in an environment with obstacles. Over the past fifteen years, many different researchers have studied sampling-based motion planning techniques such as the Probabilistic Roadmap Method (PRM) [2]. This has led to many variants, each with its own merits. It is difficult to compare the different techniques because they were tested on different types of environments, using different underlying libraries, implemented by different people on different machines.

We have provided a comparative study and analysis of a number of these techniques, all implemented in a single system and run on the same test environments and on the same computer [1]. We encountered many difficulties and pitfalls during this study. We will identify them and discuss our solutions based on our experimental research over the past four years.

2 Methods

General setup Our goal was to create a system that facilitated conducting experiments easily and reducing making errors. We met this goal by creating a single motion planning system called SAMPLE (System for Advanced Motion PLanning Experiments) which we implemented in C++ using Visual Studio.NET 2003 under Windows XP Professional.

In a motion planning experiment, many choices exist for the components that compose a sampling-based motion planning algorithm. SAMPLE provides an easy API (Application Programming Interface) to add techniques (such as a particular sampling or local planning technique) to a component and to add its parameters. We created a GUI to set up an experiment easily (see Figure 1). We considered two types of experiments. The first type compares different techniques while the second type examines the influence of a particular parameter of a technique. An experiment can be saved to/loaded from disk to enable repeating the experiment. We created a command line version of SAMPLE to run the actual experiments. This program can run on a dedicated computer that has no processes running (such as a virus scanner or an internet connection) which could influence the results. Initially, we collected the experimental data manually. As this was quite a tedious job being sensitive to copy/paste errors, we decided to automate this by automatically collecting and processing the data.

We implemented many techniques and added them to SAMPLE. Sometimes it was hard to implement a technique in a way it was intended by the creator since not all details were always present in the paper (which is often due to space limitations). In some cases, we found these details in the source code which was available on the web.

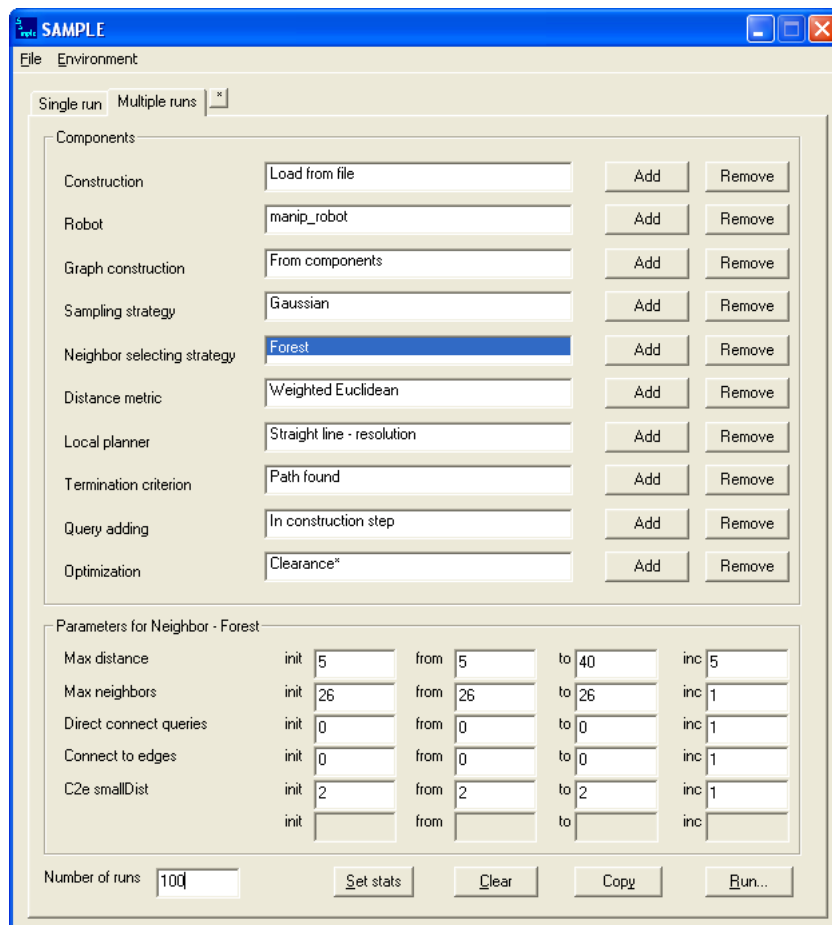


Figure 1 Setting up an experiment in SAMPLE. In this example, the effect the parameter ‘max distance’ of the ‘Forest’ neighbor selection strategy is studied. The parameter is varied between 5 and 40. For each different value, the experiment is carried out 100 times.

Representative problems The results of our comparative study indicated that some previous conclusions were too general which was probably caused by considering a set of examples that was too limited. We have to admit that it can be difficult to choose an appropriate set. In our study, we tried to employ environments and robots that resembled a wide range of configuration spaces. That is, we used environments with cluttered obstacles, narrow passages, many/few obstacles and scale differences. In addition, we used both small/large as well as different types of robots.

Some existing papers present a new technique only using examples satisfying its intentional goal. However, we think that also examples should be included that show its worst-case behavior and its limitations. For example, in our research on creating small roadmaps [1], we not only used an environment to show the potential of the algorithm, but also used an environment to show that other algorithms can be faster.

Interchangeability In our research group, initially, every member had its own implementation of his techniques which made it difficult to compare them. In addition, much work was wasted by creating components of software having comparable functionality. We decided to create libraries taking care of common functionalities such as a collision checker, a visualizer, and a graph library [5,6]. These libraries can be downloaded on the web. Also other libraries have been made available such as the Nearest neighbor library [7]. Besides the libraries, we encourage to make the source code of the complete system available such as done by [3,4].

Another important issue is the ability to exchange the geometry of environments and robots, as well as problem descriptions. We resolved this issue by using VRML as language for describing the geometry and XML as language for the descriptions.¹ There are great advantages of using existing languages: They are well-documented, parsers and type checkers are available for all appropriate platforms and programs exist to create and edit these descriptions. We think that the robotics community would benefit by supporting these languages.

3 Results

Evaluation of solution An issue each researcher has to deal with is how to evaluate the results. A common way to evaluate the results is to compare the new technique with existing techniques. In our comparative study, we initially compared techniques based on the time required to solve one relevant witness query. This however did not guarantee that every possible query could be solved by the roadmap that was constructed. We improved the study by evaluating the techniques based on solving every possible query. That is, we provided an analysis tool that indicated when the roadmap was dense enough to solve each query.

Another way to evaluate the results is to compare against the optimal solution. In some cases, we created an experiment for which the optimal solution is known. Unfortunately, such experiments can in general only be conducted for trivial cases. For more complicated experiments (such as the ones used for measuring path quality [1]), we tried to approximate the optimal solution by performing many runs. In addition, we used visual inspection to evaluate the results. In future research, we will also incorporate user evaluation to make the judgments.

¹Many geometry files can be downloaded on <http://www.give-lab.cs.uu.nl/movie/moviemodels>.

Statistics Our comparative study showed that the variance in the running times was often large. This phenomenon is undesirable because of two reasons. First, a large variation complicates statistical analysis and can even make it unreliable. Second, it is undesirable from a users point of view, e.g. it can be hard to give a user an indication of how long the method will take to terminate. Hence, we had to be very careful analyzing the results. As the running times could vary extensively, we performed a large number of runs (i.e. 100) for each experiment. In this way we increased its statistical significance. In addition, we created box plots to provide insight in the distribution of the running times. (Such a box plot displays the middle 50% of the data, the average, the standard deviation and the minimum and maximum value.)

It may seem that deterministic techniques do not have such a ‘variance problem’. Nonetheless, the study showed that a small change of the environment leads to a comparable amount of variance. Hence, care must be taken when deterministic techniques are involved.

4 Conclusion

It is often difficult to compare and evaluate techniques experimentally, because they were tested on different types of environments, using different underlying libraries, implemented by different people on different machines. By creating a system that facilitates integrating the techniques and automates conducting experiments, many errors can be avoided. To enable a fair and easy implementation of techniques, source code and software components should be made available. In addition, we encourage to use standard file formats (such as VRML and XML) to exchange problems easily.

When techniques have been implemented, they have to be evaluated by considering a large range of examples. Moreover, examples that show their limitations should also be included. A common way to evaluate the results is to compare techniques with existing ones. While such a comparison is often made based on running times, it may not always be convenient to use such a criterion. We think that incorporating user evaluation and user studies may be appropriate.

References

- [1] R. Geraerts. *Sampling-based Motion Planning: Analysis and Path Quality*. PhD thesis, Utrecht University, http://www.cs.uu.nl/~roland/motion_planning/thesis.html, 2006.
- [2] L.E. Kavraki, P. Švestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12:566–580, 1996.
- [3] J.-C. Latombe. MPK: Motion planning kit. <http://ai.stanford.edu/~mitul/mpk>, 2006.
- [4] S.M. LaValle. MSL: Motion strategy library. <http://msl.cs.uiuc.edu/msl>, 2006.
- [5] D. Nieuwenhuisen. Atlas. <http://www.cs.uu.nl/~dennis/atlas/atlas.html>, 2006.
- [6] D. Nieuwenhuisen. Callisto. <http://www.cs.uu.nl/~dennis/callisto/callisto.html>, 2006.
- [7] A. Yershova and S.M. Lavalle. MPNN: Nearest neighbor library for motion planning. <http://msl.cs.uiuc.edu/~yershova/mpnn/mpnn.htm>, 2006.

Cross-Platform Software for Benchmarks on Visual Servoing

Enric Cervera
Robotic Intelligence Lab
Jaume-I University
Castelló, Spain
Email: eervera@icc.uji.es

Abstract—The field of visual servoing is now widely accepted as a modern, consolidated discipline for vision-based real-time robot control. Since experimental setups have become affordable, many results are published worldwide in conferences and journals, yet comparison among them becomes difficult due to the wide variety of systems and tasks. A need for a common framework arise, which should allow to compare control schemes, and provide a set of benchmarking tasks to the research community. This paper presents an object-oriented, cross-platform, network-ready environment for visual servoing simulation tasks. With flexibility and extensibility as the main design goals, tasks can be defined either for cameras attached to moving Cartesian frames, or serial link manipulators. In order to provide fast feedback to the user, it includes real-time 3D rendering of the simulated scene. Task parameters and visual features can be freely chosen, and new features can be easily added to the framework. Output data is logged to disk files, which can be analyzed by any popular mathematical package. The simulator is built upon an agent-based framework, which makes possible the distribution across multiple networked computers. Moreover, it can be securely downloaded from a web server and automatically installed in a computer, running either Windows, Linux or Mac OS X operating systems, thus providing a set of common tools for a wide range of users and enabling the comparison and benchmarking of results. This simulator is now in alpha version, yet it has been extensively used by the students worldwide in an online course on visual servoing.

I. INTRODUCTION

Visual servoing is nowadays a mature yet still challenging discipline [1], [2]. Current standards in computing power along with affordable digital cameras make now possible to set up a real-time vision processing equipment capable of visually controlling a robot at frame rate. However, a lot of issues need to be tuned (internal and external calibration of the camera, feature tracking, delay in the loop, visibility constraints, manipulator workspace), thus comparison between separate works becomes increasingly difficult and the need for tools suited for benchmarking of common tasks arise.

This paper presents an object-oriented, cross-platform, network-ready environment for visual servoing simulations. With flexibility and extensibility as the main design goals, tasks can be defined either for cameras attached to moving Cartesian frames, or serial link manipulators, featuring real-time 3D visualization from the controlling camera and from auxiliary external observer cameras.

Javiss¹ (Java-based Visual Servo Simulator) departs from other simulators in that it has been designed from the scratch in a modular, distributed way. Not only the tasks are distributed among different modules (agents), but each module can run on a different computer across the network. This enables the development of distributed visual servoing tasks [3] for cooperative robot teams. In addition, such a flexible approach increases the performance of the task with distributed computing power, and it may contribute to solve hardware limitations with the real equipment.

The Javiss software is routinely used in our laboratory and it has been extensively tested by students and teachers of a pioneering online course on visual servoing [4].

Section II presents the related work on vision/robot software tools. Next, the components and technologies used in Javiss are sketched in section III. The architecture of the simulation framework is thoroughly described in section IV, along with sample results. Finally, section V outlines some conclusions and future work.

II. RELATED WORK

While hundreds of software packages for vision exist, and many other for simulated robotics are available, very few of them have been designed for both domains, thus being suitable for visual servoing tasks.

A major player since long, the Matlab Robotics Toolbox by Peter Corke [5] provides many functions that are useful in robotics such as kinematics, dynamics, and trajectory generation. It is useful for simulation as well as analyzing results from experiments with real robots. This toolbox is based on a very general method of representing the kinematics and dynamics of serial-link manipulators and models are provided for well known robots such as the Puma 560 and the Stanford arm. The companion Machine Vision Toolbox [6] provides many functions that are useful in machine vision and vision-based control.

The Matlab/Simulink Visual Servoing Toolbox [7] aimed to provide a set of functions and blocks for simulation of vision-controlled systems. The project was started during 2002 EURON Summer School on Visual Servoing, and it was developed under the supervision of the EURON Interest Group

¹<http://www.robot.uji.es/research/projects/javiss>

on Visual Servoing. Although a working version is available, its development stopped due to the limitations of Simulink for object-oriented programming. Most of the functions needed to be coded in Matlab, thus its performance was too low for real-time simulations.

Another recent Matlab extension, the Epipolar Geometry Toolbox [8] consists of a collection of functions for position and design of camera/robot and scene objects, computation of perspective/catadioptric projection and epipolar geometry (perspective and panoramic), and epipolar geometry estimation algorithms (from corresponding points) for pinhole and calibrated panoramic cameras (central).

Finally, a very important platform is ViSP (*Visual Servoing Platform*) [9], a modular framework that allows fast development of visual servoing applications. It implements the control of robot motions, the modeling of the visual features, and the tracking of the visual measurements. ViSP features a wide class of control skills as well as a library of real-time tracking processes and a simulation toolkit. The platform is a library implemented in C++, which is developed by the INRIA team Lagadic at IRISA - INRIA Rennes.

III. UNDERLYING TECHNOLOGIES

Java [10] is both a programming language and a platform. According to its designers, it is a simple, architecture neutral, object oriented, portable, distributed, high performance, multithreaded, robust, dynamic and secure high-level language.

In the simulator, real-time visualization is provided by the Java 3D API, a hierarchy of Java classes which serve as the interface to a sophisticated three-dimensional graphics and sound rendering system. Java 3D provides high-level constructs to create and manipulate 3D geometry, and to build the structures used to render that geometry. Using this API, developers can efficiently create precise virtual universes in a wide variety of sizes, from astronomical to subatomic.

Java Web Start is a deployment technology, which enables users to launch the simulator with a single click of the mouse. A version check at startup ensures that users are always up to date with the latest version. If an update is available, Java Web Start will automatically upgrade their installation.

Though a basic visual servoing setup can be made of a single computer, with a camera and a robot, our aim was to develop a framework which should be capable of managing distributed visual tasks [3]. JADE (Java Agent DEvelopment Framework) [11] is a software framework which simplifies the implementation of multi-agent systems through a middle-ware that complies with the FIPA specifications [12] and through a set of graphical tools that supports the debugging and deployment phases. FIPA is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other technologies.

IV. THE ARCHITECTURE OF JAVISS

Javiss is not only a closed simulator, but a complete developing framework for designing and implementing visual servoing tasks. Besides the description of its intuitive user

interface, this section overviews the main internal software keypoints (front-end applications, features, agents) as well as a more detailed working description of the interaction among the agents.

A. Graphical User Interface

Javiss main window and user interface is shown in Fig. 1, consisting of:

- Task properties panel, for saving and loading the parameters which completely define the current task (features, initial and desired positions, control parameters). They are stored in a text file which can be manually edited.
- Task parameters panel, which selects the choice of features, and several task parameters (gain, sampling time, error and screw thresholds) as well as the interaction matrix (at equilibrium or at current iteration). It includes two buttons to start and stop the simulation.
- Output log panel, where the user defines the names of the files where the simulation logs are stored in.
- Camera views. In the depicted front-end, the left view is provided by the camera mounted on the end-effector of the robot, while the right view is obtained from an external camera.
- Element control panel, which allows to move any object of the scene. The setup the user to select the arm (joint motion), the target object or the observer camera (Cartesian motions).

B. Design of a front-end

Modular design is a key issue in Javiss. Thus, it is not a closed simulator application but a developing framework for customizing simulations. In this way, there is a root abstract class for the application named *BasicApp* from whom the real front-ends are created. Two of them are shown as examples: *VServoCartesian* and *VServoPuma* (Fig. 2). The major part of their code is inherited from the root class, thus these applications concentrate on the design issues of each simulation task.

The code in *BasicApp* creates the window frame, allocates the necessary 3D structures, and starts the agent platform. It provides the children classes with methods for creating the different GUI elements (e.g. pose controller, joint controller or visual controller). When each of such elements is created, the corresponding agent is launched for managing the events of the GUI. In this way, the user applications are greatly simplified, since only the elements of the front-end need to be specified, with their current parameters.

Besides the GUI agents, an application consists of the objects in the scene: camera(s), robot, and target. In addition to those *physical* objects, virtual Cartesian frames can be defined too, which control the positioning of the *physical* elements attached to them.

Each object is managed by its own agent, which are created by another root abstract class named *BasicLauncher*. Each application inherits from that class the methods for creating each element, thus the user needs only to specify which

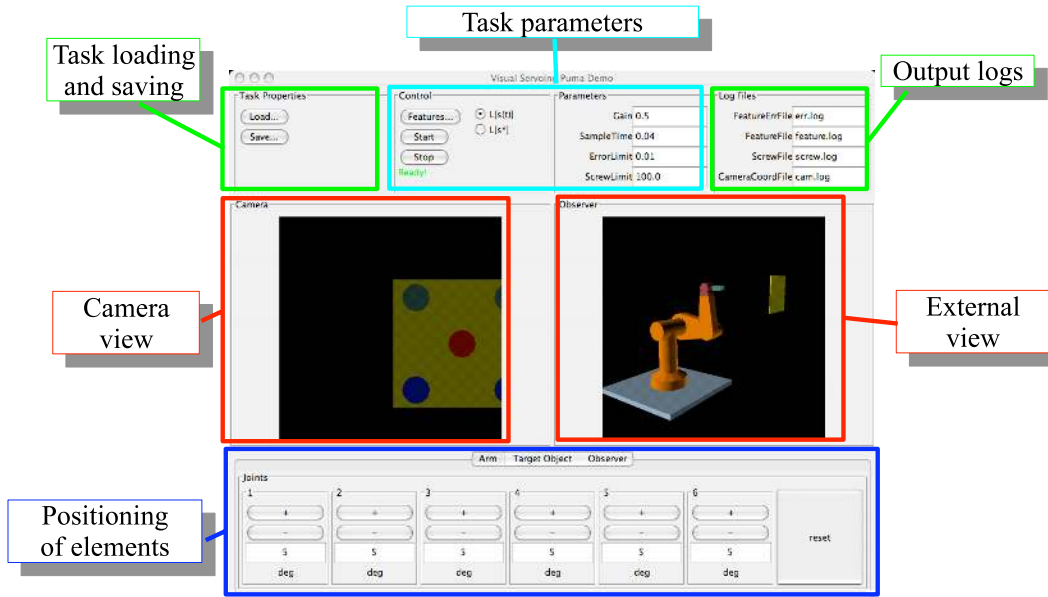
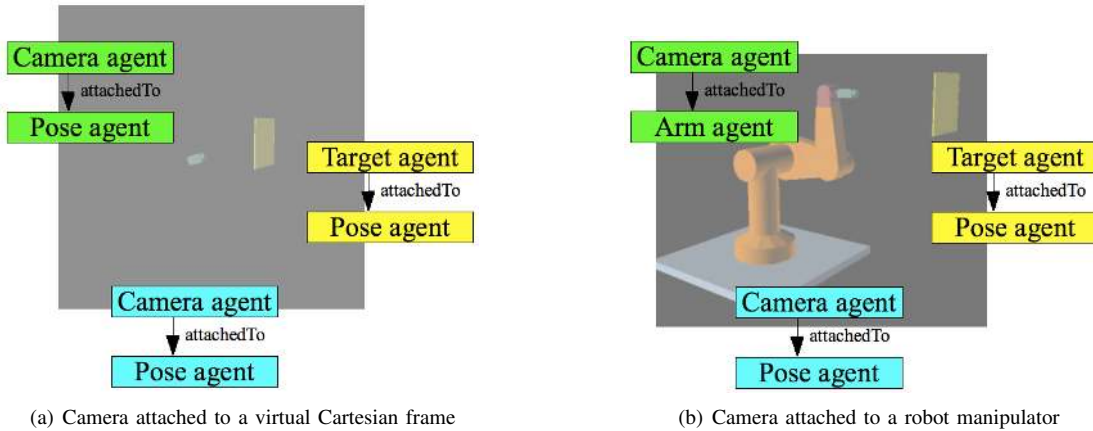


Fig. 1. Javiss main window and user interface.



(a) Camera attached to a virtual Cartesian frame

(b) Camera attached to a robot manipulator

Fig. 2. Observer view in two typical front-ends. The labels correspond to the agents that manage each element.

elements are going to be added to the scene. The role of the different agents is described in next section.

C. Agent-oriented architecture

Most modern software is designed upon the object-oriented paradigm, with emphasis on code reuse as a means of reducing the complexity of the program, and consequently the number of bugs. Agents represent a step ahead of the object model in the sense that an agent is an object which executes its own thread. Moreover, agents send and receive messages among them, which can be considered as object methods.

From an implementation point of view, agents efficiently run on distributed environments, either a network or a multiprocessor computer, thus increasing the performance of the application through parallel execution, as long as communications are effective.

In the following, the main agents playing in Javiss are briefly described. The interaction among them which leads

to the visual servoing loop will be thoroughly explained in section IV-F.

1) *Robot Agents*: The robot class is an abstract class for any serial link manipulator based on the Denavit-Hartenberg notation. It needs only to be extended for any particular robot by defining the actual values of its parameters, and its 3D CAD model for visualization. The abstract class computes the direct kinematics, and the Jacobian of the arm (based on the code of the Robotics Toolbox [5]).

2) *Pose Agent*: Objects need to be attached to a Cartesian frame. This agent manages all the operations regarding the location and motion of that frame. Target objects and cameras are attached to such agents.

3) *Camera Agent*: Besides the visualization of the scene, this agent stores the intrinsic and extrinsic parameters of the camera. It does not compute the projection of features, though. Instead, it sends the parameters to the controller agent, as described below.

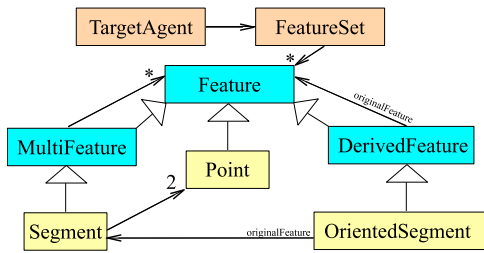


Fig. 3. UML diagram for feature classes. Light blue boxes depict abstract classes; light yellow boxes are derived example classes for point and segment features; orange boxes are top utility classes. Triangles define "is-a" (inheritance) relationships whereas plain arrows define "has-a" relationships.

4) *Target Agent*: This agent consists of the task features, and the 3D CAD model for displaying them. It also stores which features have been currently selected for the task. Upon request, it will compute the projection of those features on a given camera frame, and send the resulting information to the visual controller.

5) *Position controller Agents*: As explained before, objects can be dynamically translated and rotated by the user through the GUI. Those events are managed by these agents, which send forward messages to the corresponding Cartesian or Robot agents.

6) *Visual controller Agents*: This agent triggers the visual servo task. When the user pushes the start button on the GUI, the agent creates the log files, reads the task parameters, and then it simply sends a subscription message to the camera agent. This subscription will start the whole control loop as explained in the following.

D. Design of the feature classes

A key aspect of the design of the visual servoing framework is the treatment of features. It needs to be flexible and easy to use. To this end, several classes have been defined as depicted in the diagram of Fig. 3.

A target consists of a set of features. This set is a collection of features, which may be individual ones, multiple or derived features. For example, a point is considered an individual feature, even if it consists of three scalar parameters. In order to reuse code, a multiple feature is itself a feature, which consists of a group of simpler features. In the same example, a segment is a multiple feature consisting of two points. There is no need to define the interaction matrix for a multiple feature, since it is automatically built by stacking the interaction matrices of its composing features.

Complex features can be parametrized in different ways. A classic example is the segment, which can either be defined by two points, or by a single point, its length and its orientation. A derived feature is itself a feature again, but it references another *original* feature. The interaction matrix is computed for the original one, thus the only added computation is a derivation step as proposed in [1].

This general approach may seem cumbersome but it elegantly allows the definition of complex features based upon the aggregation of simpler ones. It is extensible, and enforces the

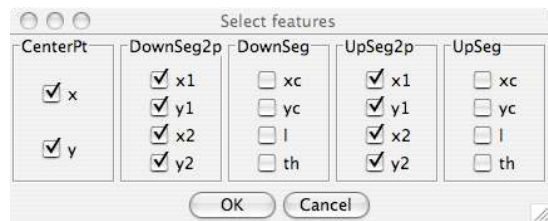


Fig. 4. Dialog for feature selection. The target consists of five points; four of them are grouped on two segments, which allows to choose either the points, or a point, the length and the orientation.

reuse of code, thus speeding and simplifying the development process.

E. Dynamic selection of features

A major issue in visual servoing is the selection of the object features which will be tracked and processed in the control law. The behavior of the task depends heavily on that choice, thus the Javiss framework allows the user to easily change her selection at runtime. A dialog (Fig. 4) displays all the available features, which can be enabled or disabled on each experiment.

The enabled features are automatically computed at each iteration, and the interaction matrix is built accordingly by stacking the rows which correspond to each feature. In this case, the target consists of five points, which can either be taken as single features, or grouped in segments (length and orientation). This illustrates a classic problem in visual servoing, when there is a high rotation around the Z axis of the camera between the initial and desired position; depending on the choice, if using point the camera will retreat (up to an infinite distance for a rotation of 180 degrees) whereas if using segment orientation it will move smoothly to the final position.

F. Visual servoing loop

There is no main control loop in the source of the simulator, but each agent has its own message loop, where it listens for incoming messages, processes them, and sends the corresponding results. By means of this model, a complete visual servoing control loop is created, as depicted in Fig. 5.

The loop involves the visual controller agent, the target agent, the camera agent, and the robot (or Cartesian) agent where the camera is attached to. When the user presses the start button of the GUI, the visual controller agent sends a subscription message to the camera agent. That means that the camera will notify the subscriber with each change in its pose.

The visual controller forwards the camera pose to the target agent, which in turn computes the projection of the selected features in the camera frame, and sends back this information to the visual controller. Now, this agent is able to compute the whole interaction matrix, and the task error vector, since the feature vector at the desired position is available at the start.

If no termination condition is triggered, the computed kinematic screw is sent back to the camera agent. This agent, however, does not move itself, but it sends forward the screw

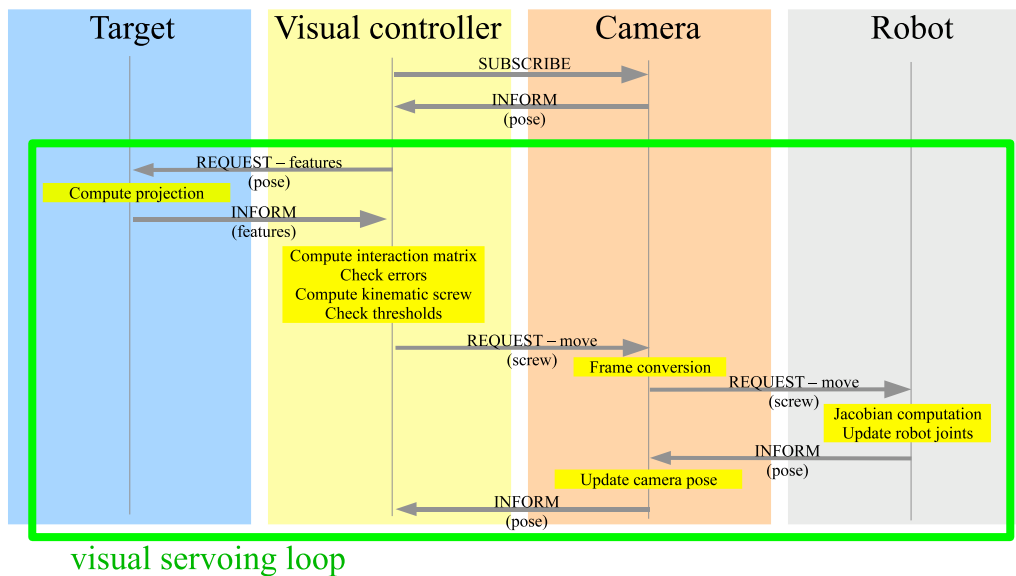


Fig. 5. Flow of messages among agents in the visual servoing loop. The visual control agent triggers the task by subscribing to the camera agent. Then, the information flows through messages among the agents, as shown in the picture, until it is stopped either by the user, or by any termination condition.

(with the appropriate frame conversion) to the robot agent where the camera is attached. This last agent will compute the Jacobian matrix, and update the joint values accordingly.

Finally, the updated pose of the robot is sent to the camera, then converted to the camera frame, and sent again to the visual controller, and the loop starts a new iteration.

This is a very flexible framework, since the agents do not need to run on the same platform, and it behaves similarly to a distributed visual controller which has been implemented in a real setup [3].

G. Output logs

Javiss does not include any plotting or displaying of the resulting task data. Instead, all the information is stored in log files, in ASCII format. The data can then be retrieved by any popular plotting or mathematical package such as Matlab or *gnuplot* for displaying, as seen in Fig.6 (sample scripts are provided in the website for plotting each log file using the free tool *gnuplot*).

Filename can be changed dynamically prior to task execution, otherwise they will be overwritten. The application logs the task error $s - s^*$, the feature vector s , the kinematic screw v and the position of the camera (as an homogeneous matrix).

H. Deployment, installation, and updating

Software packages are nowadays widely distributed via Internet. The usual cycle consists of downloading, compiling, and installing. The compiling process may be a pain if the software has been designed with a particular operating system and a set of libraries in mind, even for specific versions.

The Java platform offers the Web Start technology which automatically downloads and installs the software from a web page. It does not need to be compiled, since the Java bytecode runs on every platform, and all the dependencies

are automatically loaded too. Moreover, if new versions are available when the user executes the application, the system automatically downloads and installs them in a transparent way. Downloaded software may be digitally signed by trusted certificates, thus ensuring that the software does not contain any virus nor spyware.

The efficiency of such a deployment mechanism has been demonstrated in the Online Course on Visual Servoing [4]: teachers and students were able to install Javiss either on Windows or Linux without significant difficulties. Most of them had little or no previous experience with Java, nor any special computer administration skills. They were able to download and begin to execute Javiss in a couple of days, and the main encountered problems dealt with the installation of Java on the operating system.

V. CONCLUSION

A visual servo simulator has been presented, which enables the easy experimentation of visual control tasks on a wide variety of platforms and setups, thus becoming a powerful yet easy to use benchmarking tool. Compared to similar available software, Javiss sets apart in its distributed, object-oriented design. Its agent-based philosophy makes it directly usable in networked environments, as well as making full use of distributed computers and multiple processors.

A great effort was put in the design of the feature classes, to enable the user to easily derive new feature objects based on existing ones.

Cross-platform languages and technologies enable a robust and widespread use of the program, which is easily deployed and installed via the World Wide Web.

While the software has been extensively tested by students and teachers of an online visual servoing course [4], new options and enhancements are on the way. More realistic

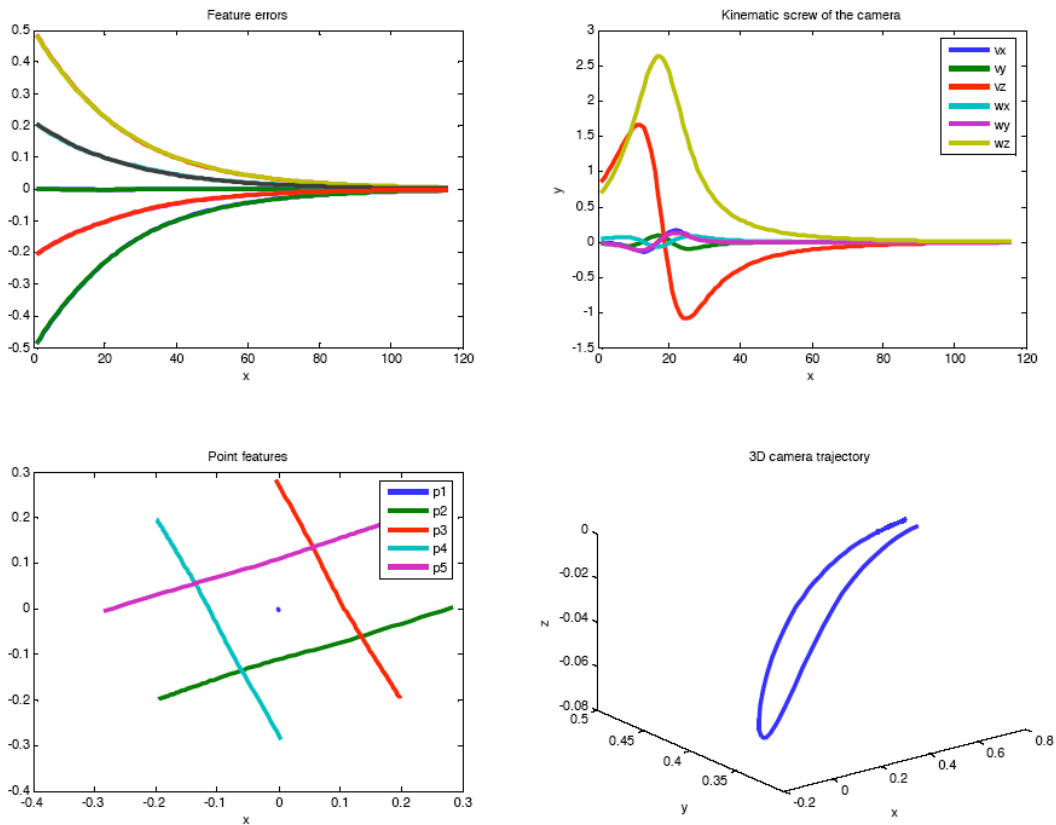


Fig. 6. Data plots in Matlab based on the Javiss log files. Left-up: task feature errors; left-down: task features as observed by the camera; right-up: kinematic screw; right-down: 3D trajectory of the camera, where the retreat motion can be observed.

simulations can be achieved by incorporating the dynamics of manipulators [5]. Stereo and multicameras could be managed with the existing components, but a more general arbitration scheme for the computation of the interaction matrix needs to be worked.

Finally, we plan to combine simulation and real devices (cameras and robots) in the way that the same code which runs on the simulator can be tested on real equipment, by using the native interface capabilities of the Java platform. Therefore, benchmarks could be extended to real robotic systems, still a challenge nowadays.

ACKNOWLEDGMENT

The author would like to thank Peter Corke for granting the use of part of his code from the Robotics Toolbox, the support by the Spanish Ministry of Science and Education under grant DPI2005-08203-C02-01, and the valuable comments and feedback of François Chaumette, Ezio Malis, Philippe Martinet, and the rest of teachers and students of the International Online Course on Visual Servoing Techniques.

REFERENCES

[1] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Trans. Robot. Automat.*, vol. 8, no. 3, pp. 313–326, June 1992.

[2] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Trans. Robot. Automat.*, vol. 12, no. 5, pp. 651–670, 1996.

[3] E. Cervera, "Distributed visual servoing: A cross-platform agent-based implementation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, 2005, pp. 3676 – 3681.

[4] International Online Course on Visual Servoing Techniques. <http://www.robot.uji.es/IOCoViST/>.

[5] P. I. Corke, "A robotics toolbox for MATLAB," *IEEE Robot. Automat. Mag.*, vol. 3, no. 1, pp. 24–32, March 1996.

[6] —, "The machine vision toolbox - a MATLAB toolbox for vision and vision-based control," *IEEE Robot. Automat. Mag.*, vol. 12, no. 4, pp. 16–25, March 2005, special Issue on Software Packages for Vision-Based Control of Motion.

[7] E. Cervera, "Visual servoing toolbox for MATLAB/Simulink," 2002, <http://vstoolbox.sourceforge.net>.

[8] G. Mariottini and D. Prattichizzo, "EGT: a toolbox for multiple view geometry and visual servoing," *IEEE Robot. Automat. Mag.*, vol. 12, no. 4, pp. 26–39, December 2005, special Issue on Software Packages for Vision-Based Control of Motion.

[9] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing," *IEEE Robot. Automat. Mag.*, vol. 12, no. 4, pp. 40–52, December 2005, special Issue on Software Packages for Vision-Based Control of Motion.

[10] Java Technology. <http://java.sun.com>.

[11] Java Agent Development Framework. <http://jade.tilab.com>.

[12] Foundation for Intelligent Physical Agents. <http://www.fipa.org>.

Drawing a parallel: Benchmarking in Computer Vision and Robotics

Danica Kragic*, Ville Kyrki†, Patric Jensfelt* and Frank Lingelbach*

* Centre for Autonomous Systems,
Royal Institute of Technology, Stockholm, Sweden,
dani, patric, frankl@kth.se,

† Laboratory of Information Processing,
Lappeenranta University of Technology, Lappeenranta, Finland,
kyrki@lut.fi

1 Introduction

Since early 1990s there has been a growing interest in the computer vision community on discovering ways to compare the performance of different methods. In this paper, the current state of benchmarking and performance comparison in vision is reviewed, with the goal of discovering good benchmarking practices that could be also introduced in the robotics community. In short, what can researchers in robotics learn about benchmarking by taking a look at computer vision benchmarking procedures?

An important common characteristic of both robotics and computer vision is that both are highly hardware and application dependent, and therefore many similar problems exist even though “pure” computer vision has still somewhat less variation. In both fields the tasks to be achieved are complex, making analytical performance prediction impossible in many cases thus leaving empirical study as the only available approach. For this reason, the test cases of the empirical studies, as well as the analysis of the results of experiments, are the most important considerations in benchmarking.

2 Benchmarking in Vision

The increased interest to benchmarking in computer vision during the last decade can be easily seen in a number of projects concentrating on benchmarking. For example, the EU funded PCCV (Performance Characterization in Computer Vision) project produced tutorials and case-studies for benchmarking vision algorithms [17].

2.1 Types of tools

Three main tools of performance evaluation in computer vision are a) common/benchmark data sets, b) workshops of performance evaluation, and c) contests. The goal of common data sets is usually to provide a base for comparing

different approaches, and nowadays such data sets are available for most of the important application areas of computer vision. For example, FERET and XM2VTS databases for face recognition [18, 14], COIL databases for object recognition [16] and CURET database for textures [3]. A very important development on the benchmark data is the introduction of standardized test policies for the data sets. This inherently makes the results achieved by different researchers comparable to each other. Common test policies have become important especially in biometric person authentication. That is, there are now some well known data sets and test policies which provide a baseline for comparison of different methods. This means that a new approach has to be competitive at least on the basic data to be considered seriously by the community.

Workshops in performance evaluation, such as the highly successful series of PETS (Performance evaluation in tracking and surveillance) workshops, aim to collect together researchers working on a particular problem to discuss both the performance of individual methods and the performance evaluation criteria. For example, in PETS workshops, a common data set is published before the workshop by the organizers, and each individual method presented must be tested with the common data. The data provided is often divided to training and test sets, but the whole data is usually available for method development. However, no universal evaluation criteria is usually presented and no best method is selected, in contrast to contests. Workshops contribute also by directing research on challenging future topics.

Different kinds of contests have recently become popular in evaluation of computer vision algorithms. The range of contests goes from academic, such as the ICCV 2005 contest on location recognition, to industrial, such as the NIST organized Face recognition vendor tests. In contests, the precise evaluation criterion is almost always given in advance, with some training data. However, the contest is usually performed using data not available to contestants before the contest. One recognized problem of contests is that they can make a single algorithm “the standard”, which might limit research on other approaches, even if the algorithm is not “universally good”, that is, it has some pitfalls, known or unknown.

A way to avoid the above problem is to base the evaluation on a strawman, i.e., comparing to a well-known viable solution which does not need to be state-of-art (e.g., Canny edge detector). Anything working better than the strawman might warrant publication. This also eases the comparison between different algorithms, as the strawman methods are usually generally available, while the state-of-art approaches might not be.

2.2 Technology vs application evaluation

There is some debate over using real versus artificially generated data in performance evaluation. This debate can be viewed by considering the differences between *technology evaluation* and *application evaluation*. In technology evaluation, an algorithm’s response is evaluated with respect to factors such as its tuning parameters or amount of particular type of noise in input data. In this evaluation, simulated data is very useful as e.g. the noise amount can be controlled. In application evaluation, the performance is evaluated in a particular task, which means that real (application) data is needed. An important note here is that having application knowledge and correct evaluation of technology

can be used to predict the application performance, but the unexpected application phenomena can never be evaluated without real application data. This two-fold division corresponds to the use of simulations versus real-life experiments in robotics.

2.3 Levels of benchmark data

The level of standardization in common data sets can be divided into three levels: a) common data without ground truth, b) data with ground truth, and c) data with test policy. Sometimes having common data is useful even if there is no associated ground truth. One reason for not having the ground truth is that in several higher level problems, it is very difficult to determine an unambiguous ground truth. This might be the case, for example, in surveillance scenarios where the scene would need to be annotated by a human expert causing ambiguities and limited resolution of the ground truth. Common data without ground truth is not directly a tool for performance evaluation, while it can still be useful to direct research on a certain topic.

Common data with ground truth but without a strict test policy is the most common type of benchmark data. It allows some benchmarking, but a typical problem is that the performance measures used are not strictly defined. Thus, comparison of results requires often implementation of all methods compared in order to ensure that the same performance measure is used. In computer vision, many methods are based on learning based on examples. In this case it is essential that the benchmark data is clearly divided into separate subsets for learning and testing. Finally, it should be noted that defining the ground truth is not always unambiguous, for example, in object category recognition where the categories are determined by a human expert.

The most complete level of benchmark data includes the test procedures and performance measures in addition to the actual data. This kind of benchmarking is very useful because the results are directly comparable and replication of other methods is not necessary. This approach has become very important in biometrics, where the scenarios as well as the ground truth can be precisely defined. However, in some other areas with multiple and less precise performance measures, it is less useful. The use of common test policy is especially difficult in active vision, where the vision system has some control over its input data, and in this case is only possible through simulations.

2.4 Measuring performance

We will now briefly overview the types of performance metrics widely used in computer vision. There are two basic types of performance measures, depending on if the output of the method in question is discrete or continuous. If the method output can be evaluated as either correct or incorrect, the performance is typically evaluated using discrete error counting. However, in many cases there are two different failure modes, false positives and false negatives, and these two may have different effect in an application. For example, in person authentication false positives might be more harmful than false negatives. In this case, the output of a method typically depends on a threshold value, which controls the ratio between false positives and false negatives. ROC (receiver operating characteristic) is a graphical method of describing the performance

over all values of the threshold. It is also possible to compare two methods by comparing their ROC curves.

For methods with continuous output, the typical approach is to determine the MSE (mean square error) with respect to the ground truth. For cases where the output errors are small and approximately normally distributed, this is a good measure. However, if the method can sometimes even rarely break down totally, its MSE is very high even if in most cases its error remains low. This is due to the unseen assumption of normal distribution underlying the measure. In addition to the MSE error, the distribution of errors should be described for example by the variance of errors, which gives some outlook on the distribution.

The performance values should always be examined using standard statistical techniques independent of the type of the performance metric. For example, hypothesis testing can be used to determine if the difference in performance between two methods is statistically significant, taking into account the sample size available.

3 Benchmarking in Robotics

The area of robotics is very wide and includes a large range of research fields. That this is the case is evident when studying the list of sessions or the topics of interest mentioned in the CFP for one of the major robotics conferences such as IROS and ICRA. An incomplete list of these areas include: manipulation, obstacle avoidance, planning, humanoids, hardware design, SLAM, vision, sensors, teleoperation, learning, Vision is thus just one of many topics in robotics. Many of the areas also have subdomains just like vision has (object recognition, tracking, . . .) and some of the areas are intimately connected. To speak of performance in robotics in general is therefore not possible. Thus, this paper will take a look at some of the domains within robotics.

One aspect of some of the robotics problems which does not appear in mainstream computer vision is the active control aspect. Active control of sensors and actions does not allow for easy performance evaluation on data sets.

In some cases, but not all, simulation provides a way to at least repeat experiments with exactly the same and well known environmental conditions. The problem with simulation is that it is only as good as the simulation model and typically never fully captures the complexity of the real world.

There have been a number of successful contests within robotics. Some like the one at the AAAI conference has been running for a long time. Recently the DARPA Grand Challenge generated a lot of media attention and the RoboCup is also something that many outside of the community has heard about.

- The 12th annual AAAI Mobile Robot Competition will be held this year and consists of three events related to mobile robots
<http://robots.net/rcfaq.html>
- RoboCup that has as a longterm goal to develop a robot soccer team that will beat human world champions
<http://www.robocup.org/>
- DARPA Grand Challenge -
<http://www.darpa.mil/grandchallenge/index.asp>

- ELROB - 1st European Land-Robot Trial
<http://www.elrob2006.org/>

Workshops are also being organized on performance evaluation in robotics such as the one in conjunction with the EUROS2006 conference.

As was already discussed in the context of vision, there is a need for available baseline methods when evaluating new robot applications. It is quite common that researcher only provides a comparison of the new results with his last result and that of his group. Having a set of available methods to compare to would advance the field. There is some code available but comparisons are made difficult because the hardware is typically different as well.

3.1 SLAM

One key competence for a fully autonomous mobile system is the ability to build a map of the environment from sensor data and use it to perform various tasks such as navigation and localization. The field of Simultaneous Localization and Mapping (SLAM) has matured significantly over the last years. Systems based on laser scanners have shown impressive performance [20, 2, 6], especially in indoor environments. Recently, much of the attention has shifted towards using vision as the main sensor [4, 8, 19, 12] due to its low cost and power consumption, richness of information and potential of retrieving 3D structure.

In terms of performance evaluation in SLAM, there are now a number of sites on the web where data sets can be downloaded (see [5] for a list). This is an important step to be able to compare results of different methods with the same input. Some, like the Victoria Park data set from Eduardo Nebot at the University of Sydney has become what the COIL data base was for object recognition and appear in many SLAM publications. There are also some SLAM software available online (see [5]).

In terms of performance evaluation execution time and computational complexity have been the two most quantitative measures so far. The size of the environment that a certain algorithm could handle and still remain consistent has also been used but it is not until the same data sets has been used that this has been a really useful measure. A problem with using the common data sets is that the same set is typically used for parameter tuning and evaluation, that is, the parameters of the algorithm are adapted to make the results as good as possible on a certain data set as opposed to tuning for one and then running on another as would be the proper experimental evaluation procedure.

What is still missing is a generally accepted metric for evaluating the quality the generated map on any of the many available data sets. To use an extreme example, how does one compare the result of two SLAM algorithms if one builds a metric map and the other a topological map? Evaluating the quality would require having some kind of ground truth which in anything but a toy environment or simulation is a staggering task.

One thing that would be possible to measure and compare to ground truth would be the position estimate that a SLAM algorithm produces. There are plenty of accurate localization systems reported in the literature which could be used to gather the ground truth position data.

3.2 Obstacle Avoidance

Obstacle avoidance is another key competence in a mobile robot system. The robot must be able to move about without colliding with the environment. This is an example of an area where active control is an integral part of the problem, given sensor data how to control the robot so that it avoids the obstacles but still reaches the goal.

In obstacle avoidance the algorithm cannot be evaluated in the real world decoupled from the implementation. That is, the performance of the algorithm will depend heavily on how well it is implemented. Execution time, control frequency, delays, filtering of sensor noise are all examples of factors that have to be dealt with but are typically not integral to the algorithm itself.

One way to make quantitative evaluation, suggested by Javier Minguez during a workshop on benchmarking at the EUROS 2006 conference, is to use a simulator in which different methods can be evaluated with the same environmental conditions for a large number of different obstacle configurations and without many of the implementation issues. This would be an example of a technology evaluation. As obstacle avoidance by nature is tightly coupled to a real implementation an application evaluation is also needed as well.

Evaluations of new methods in obstacle avoidance literature consist, in some cases, in comparing to one of a couple of standard methods. Such a comparison is however typically not quantitative but rather qualitative and not on the same test scenario but rather compared to experiments performed by others with different hardware, software and environmental conditions. Some of the most common methods to compare against are potential field methods [13], the Vector Field Histogram [1], the Dynamic Window Approach [7] and the Nearness Diagram [15]. Even though obstacle avoidance and in particular the subproblem of obstacle detection is not solved the research activity has slowed down somewhat lately.

3.3 Visual Servoing

Visual servoing has been recognized as one of the core robotics subareas in which it is possible to define different benchmarking procedure, [9]. Visual servoing in general consists in specifying a control task as the regulation of a position/pose of visual features/objects. One of the scientific questions that is suitable for benchmarking is the design of the control strategy. In relation, time to convergence, the steady state error and the distance traveled may be three evaluation criteria used for benchmarking. The existence of visual servoing simulation environments facilitate this in the near future [10, 11].

3.4 Path Planning

Path planning is a field that, at a first glance, seems to be perfectly suitable for benchmarking. There is no robotics hardware involved but a computer and no sensor information has to be processed. A path planning problem in robotics consists of a world model, a model of the robot and a start and a goal configuration. All a path planning algorithm has to do is to find a collision-free path from start to goal. Thus, having a database of problems would clearly enhance the comparability of work on path planning algorithms. Unfortunately,

such a database does not yet exist. Instead, research groups test their algorithms on their own sets of benchmark problems. Even if many of these problems are available on the internet, the path planning community has not yet agreed on a common set.

Path planning problems come in a number of different shapes. Of course, there are models of real robots acting in models of work cells or home environments. In addition, there are artificial problems that explicitly model specific difficulties like variants of the narrow passage problem. The (multi) rigid body problem is very common, where the *robot* is simply a free flying rigid body. For example, the original piano-movers problem is of such kind. Robotics is only one field of application for path planning methods. Other applications – and thereby other *natural* problems – exist, for example, in computational biology or assembly planning. Using the configuration space approach, the different appearances reduce to the problem of finding a path for a point-like agent in the configuration space. Thus, all the different variations of path planning problems can be solved by an algorithm capable of finding a path for this point-like agent. The information about the geometry of a robot and the world surrounding it is hidden in a binary collision checker. For a given configuration, this function computes the image of the robot in the workspace and checks whether it collides with an obstacle or not.

The most important measure on how well a path planning algorithm performs is computation time. Obviously, comparing computation time is difficult considering different software packages written in different programming languages running on computers with different configurations. A second measure is the number of collision checks an algorithm needs to solve a path planning query. Many sampling based methods like PRM or RRT spend the majority of computation time on collision checking. But comparing the number of collision checks can neither be an objective measure. It would favor methods like PCD that spend a considerable amount of time on maintaining data structures to intelligently choose configurations to be checked for collision. The quality of a path is rarely an issue and, if so, mostly covered by a subsequent optimization step. Optimality can be defined in a number of different ways. In robotics, the length of the path is certainly relevant. But if the path is to be followed by a real robot, the robustness of the path – in terms of keeping a safety distance to obstacles – might be more important.

Computation time will most probably remain being the main measure in path planning. Therefore, not only a common set of benchmark problems but also a common planning environment seem to be the keys to obtain objective benchmarks in path planning.

4 What can Robotics learn from Computer Vision

- Generating databases helps! But in addition to data, also performance metrics and test procedures are needed.
- Solving hardware curse problem: Necessary? Complete solution to hardware dependency is not likely to be possible. However, the available benchmark data sets should contain real-world data instead of pure simulation. There

are still no good solutions to hw problem in active vision either. A good way to evaluate performance on complex tasks is most likely to follow the ideas of DARPA Grand Challenge, which leaves most of the options open for the participants.

- Simulation environments: Needed? Definitely yes, because in many cases they are the only objective way of comparing algorithms. Of course, they only allow technology evaluation, which might nevertheless be valuable.
- Open source: Needed? At least freely available baseline methods should be found, even if their source code would be closed.

References

- [1] J. Borenstein and Y. Koren. The vector field histogram - fast mobile obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, June 1991.
- [2] Michael Bosse, Paul Newman, John Leonard, and Seth Teller. SLAM in large-scale cyclic environments using the Atlas framework. *International Journal of Robotics Research*, 23(12):1113–1139, 2004.
- [3] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink. Reflectance and texture of real-world surfaces. Technical Report CUCS-048-96, Columbia University, 1996.
- [4] Andrew J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc. of the ICCV*, 2003.
- [5] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: Part I. *IEEE Robotics & Automation Magazine*, 13(2):99–108, 2006.
- [6] C. Estrada, J. Neira, and J.D. Tardós. Hierarchical SLAM: Real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 2005.
- [7] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, March 1997.
- [8] Luis Goncavles, Enrico di Bernardo, Dave Benson, Marcus Svedman, Jim Ostrovski, Niklas Karlsson, and Paolo Pirjanian. A visual front-end for simultaneous localization and mapping. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'05)*, pages 44–49, Barcelona, Spain, apr 2005.
- [9] <http://www.euron.org/activities/benchmarks/index.html>.
- [10] <http://www.irisa.fr/lagadic/visp/visp.html>.
- [11] <http://www.robot.uji.es/benchmarks/competition/visual.html>.

- [12] Patric Jensfelt, Danica Kragic, John Folkesson, and Mårten Björkman. A framework for vision based bearing only 3D SLAM. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'06)*, Orlando, FL, May 2006.
- [13] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'85)*, pages 500–505, St. Louis, MO, March 25–28, 1985. IEEE.
- [14] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre. Xm2vtsdb: The extended m2vts database. In *Second International Conference on Audio and Video-based Biometric Person Authentication*, March 1999.
- [15] J. Minguez and L. Montano. Nearness diagram navigation (nd): A new real time collision avoidance approach. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'00)*, pages 2094–2100, Takamatsu, Japan, 2000.
- [16] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-100). Technical Report CUCS-006-96, Columbia University, 1996.
- [17] PCCV. Performance characterization in computer vision website. <http://peipa.essex.ac.uk/benchmark/index.html>.
- [18] P. J. Phillips, H. Moon, P. J. Rauss, and S. Rizvi. The feret evaluation methodology for face recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10), 2000.
- [19] Juan Solà, André Monin, Michel Devy, and Thomas Lemarie. Undelayed initialization in bearing only SLAM. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'05)*, pages 2751–2756, August 2005.
- [20] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'00)*, volume 1, pages 321–328, San Francisco, CA, USA, 2000.

