

# Experiments in Real-time Vision-based Point Stabilization of a Nonholonomic Mobile Manipulator

D.P. Tsakiris, K. Kapellos, C. Samson, P. Rives and J.-J. Borrelly  
Project Icare, INRIA, U.R.-Sophia Antipolis,  
2004, Route des Lucioles, BP 93,  
06902 Sophia Antipolis Cedex, France  
*{first\_name}.{last\_name}@sophia.inria.fr*

## Abstract:

The stabilization to a desired pose of a nonholonomic mobile robot carrying a manipulator arm, based on sensory data provided by a camera mounted on the end-effector of the arm, is considered. Instances of this problem occur in practice during docking or parallel parking maneuvers of such vehicles. The visual data obtained as the camera tracks a target of known geometry are used to implement a continuous time-varying state feedback for the stabilization of the mobile robot. We focus on the case where the camera moves parallel to the plane supporting the mobile robot. The experimental evaluation of the proposed techniques uses a mobile manipulator prototype developed in our laboratory and dedicated multiprocessor real-time image processing and control systems. The real-time programming aspects of the experiments are handled in the context of the ORCCAD control architecture development environment.

## 1. Introduction

We consider here the experimental evaluation of certain recently-developed techniques for the stabilization to a desired pose of a mobile manipulator composed of a nonholonomic wheeled unicycle-type vehicle on which a holonomic manipulator arm is mounted (Tsakiris, Samson and Rives [18], [19]).

The presence of nonholonomic constraints makes this task non-trivial. These constraints arise from the rolling-without-slipping of the mobile platform's wheels on the plane supporting the system and constrain its instantaneous motion, whose component lateral to the heading direction has to be zero.

The kinematics of such nonholonomic mobile robots can be modeled as drift-free controllable nonlinear systems with fewer controls than states, with the controls entering linearly in the state equations. One of the approaches developed to solve the stabilization problem for such systems is the use of time-varying state feedback, i.e. control laws that depend explicitly, not only on the state, but also on time, usually in a periodic way. Samson [14] introduced them in the context of the unicycle's point stabilization and raised the issue of

the rate of convergence to the desired equilibrium. M’Closkey and Murray [6], [7], Pomet and Samson [11] and Morin and Samson [8] derived continuous non-Lipschitz periodic time-varying exponentially stabilizing controls, which make the closed-loop system homogeneous of degree zero.

The implementation of such a closed-loop control scheme requires the state of the mobile robot to be estimated at every time instant. In our work, we use visual data from a camera which is mounted at the tip of the manipulator arm. The extra degrees-of-freedom associated with the arm, make it possible to position the end-effector, and thus the camera, independently from the mobile platform using a visual servoing approach. In this way, the camera is able to track a target of interest, enabling the localization of the system with respect to this target, while the nonholonomic mobile platform performs the maneuvers necessary to its own positioning.

For static manipulator arms, the visual-servoing approach provides a way of accomplishing such vision-based stabilization tasks by introducing directly visual feedback in the system’s control loop (Espiau, Chaumette, Rives [2], Hashimoto [5], Hager and Hutchinson [4]). To use this approach for mobile manipulators, we extend it to the case of mobile robotic systems with nonholonomic motion constraints. Previous work in this area by Pissard-Gibollet and Rives [9] shows how to position a camera mounted on a nonholonomic mobile robot in front of a given target, without, however, explicitly controlling the final position and orientation of the mobile robot.

The time-varying stabilizing control scheme presented in [18] uses the unicycle’s absolute position and orientation, which are reconstructed from the visual data. Alternatively, a scheme is proposed in [19] where the visual data enter directly in the control loop, without the intermediate step of state reconstruction. This gives rise to a visual-servoing scheme for stabilizing the nonholonomic mobile manipulator to a desired configuration, the first such scheme, to our knowledge, to appear. Details are presented in section 2.

The software implementation of the control and sensing schemes for our experiments and the treatment of the corresponding real-time aspects need to be done in a coherent way. To do that, we use the ORCCAD control architecture design environment (Simon, Espiau, Castillo and Kapellos [16], [3]), as explained in section 3.

Our experiments were carried out using a mobile manipulator prototype developed at INRIA. Initial experiments using odometry and the state reconstruction-based method of [18] agreed with the results of the corresponding simulation studies. Experiments in progress demonstrate the validity of the visual servoing method of [19], focusing on its comparison with the previous technique and on improving its robustness in the presence of noise, delays and parameter uncertainties. Details on the experiments are given in section 4.

## 2. Vision-based Point Stabilization

Let  $(x, y)$  be the position of the midpoint  $M$  of the wheel axis and  $\theta$  be the orientation of the mobile robot with respect to an inertial coordinate system coinciding with the desired equilibrium configuration in point  $O$ . We consider

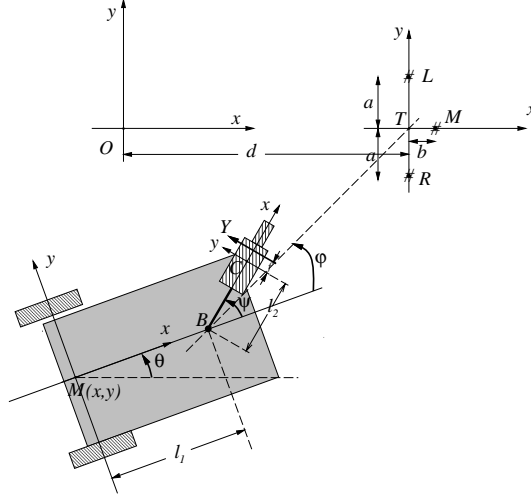


Figure 1. Unicycle with Camera

the unicycle kinematic model with heading speed  $v$  and angular velocity  $\omega$  :

$$\dot{x} = v \cos \theta, \quad \dot{y} = v \sin \theta, \quad \dot{\theta} = \omega. \quad (1)$$

Let  $\psi$  be the angle of the manipulator arm with respect to the body of the mobile robot and let  $\omega_\psi$  be the corresponding angular velocity. Then

$$\dot{X} = B_3(X) (v, \omega, \omega_\psi)^\top, \quad (2)$$

where  $X \stackrel{\text{def}}{=} (x, y, \theta, \psi)^\top$  is the system state and  $B_3(X)$  can be easily specified from equation 1.

Consider also a fixed target containing three easily identifiable feature points arranged in the configuration of fig. 1, which we suppose to be at a distance  $d$  from the point  $O$ . The distances  $a$  and  $b$  (fig. 1) are assumed to be known. The coordinates of the three feature points with respect to the camera are  $(x_p^{\{C\}}, y_p^{\{C\}})$ ,  $p \in \{l, m, r\}$  and their corresponding coordinates on the (1-dimensional) image plane are  $Y_p$ ,  $p \in \{l, m, r\}$ .

From the system kinematics and the perspective projection camera model, we get the mapping  $Y = \Phi(X)$  between the system state  $X$  and the sensory data  $Y \stackrel{\text{def}}{=} (Y_l, Y_m, Y_r, \psi)^\top$ . The corresponding Jacobian is  $J(X) \stackrel{\text{def}}{=} \frac{\partial \Phi}{\partial X}(X) = B_2(X) B_1(X)$ , where the matrices  $B_1(X)$  and  $B_2(X)$  are given in [19]. The matrix  $B_2(X)$  corresponds to the interaction matrix of [2], [9]. The dimensions  $l_1$  and  $l_2$  of the mobile robot are shown in fig. 1 and  $f$  is the focal length of the camera.

The problem that we consider is to stabilize the unicycle to the desired configuration  $X_* = (x_*, y_*, \theta_*, \psi_*)^\top = 0$ . The corresponding sensory data  $Y_* = (Y_{l*}, Y_{m*}, Y_{r*}, \psi_*)^\top = \Phi(0)$  can be directly measured by putting the system in the desired configuration or can be easily specified, provided  $d$  is also known, along with the target geometry  $a$  and  $b$ .

An exponentially stabilizing control is considered for the unicycle, while a control that keeps the targets foveated is considered for the camera.

The following state transformation brings equations 1 in the so-called *chained form* [11], [8]:

$$(x_1, x_2, x_3)^\top = \Psi(X) \stackrel{\text{def}}{=} (x, y, \tan \theta)^\top. \quad (3)$$

The unicycle control, that can be used if the state is known or reconstructed, is given by:

$$v(t, X) = u_1(t, \Psi(X)) / \cos \theta, \quad \omega(t, X) = \cos^2 \theta u_2(t, \Psi(X)) \quad (4)$$

where  $u_1$  and  $u_2$  are the time-varying state-feedback controls, developed by Morin and Samson [8] for the 3-dimensional 2-input chained-form system and which are given in terms of the chained-form coordinates of equation 3 by:

$$\begin{aligned} u_1(t, x_1, x_2, x_3) &= k_1 [\rho_3(x_2, x_3) + \alpha(-x_1 \sin wt + |x_1 \sin wt|)] \sin wt, \\ u_2(t, x_1, x_2, x_3) &= -\frac{k_3}{\rho_3(x_2, x_3)} \left[ |u_1| x_3 + k_2 u_1 \frac{x_2}{\rho_2(x_2)} \right], \end{aligned} \quad (5)$$

where  $\rho_2(x_2) \stackrel{\text{def}}{=} |x_2|^{\frac{1}{3}}$ ,  $\rho_3(x_2, x_3) \stackrel{\text{def}}{=} (|x_2|^2 + |x_3|^3)^{\frac{1}{6}}$ ,  $w$  is the frequency of the time-varying controls and  $\alpha, k_1, k_2, k_3$  are positive gains. The exponential convergence to zero of the closed-loop system can be demonstrated using the homogeneous norm  $\rho(x_1, x_2, x_3) \stackrel{\text{def}}{=} (|x_1|^6 + |x_2|^2 + |x_3|^3)^{\frac{1}{6}}$ .

The arm control  $\omega_\psi$  is chosen to keep the targets foveated by regulating the angular deviation of the line-of-sight of the camera from the targets to zero, while the mobile robot moves. It is specified so that  $Y_m$  is made to decrease exponentially to  $Y_{m\star}$ , i.e. by making the closed-loop system for  $Y_m$  behave as  $\dot{Y}_m = -k_4(Y_m - Y_{m\star})$ , where  $k_4$  is a positive gain. This gives

$$\omega_\psi(t, X, Y) = -\frac{k_4}{\mathcal{J}_{2,3}}(Y_m - Y_{m\star}) - \left( \frac{\mathcal{J}_{2,1}}{\mathcal{J}_{2,3}} v + \frac{\mathcal{J}_{2,2}}{\mathcal{J}_{2,3}} \omega \right), \quad (6)$$

where  $\mathcal{J}_{2,i}$  is the  $(2, i)$ -entry of the  $4 \times 3$  matrix  $\mathcal{J}(X) \stackrel{\text{def}}{=} B_2(X)B_1(X)B_3(X)$ . In particular,  $\mathcal{J}_{2,3} = -f - \left( \frac{Y_m^2}{f} + \frac{l_2 f}{x_m^2 c_1} \right)$ . The first term of equation 6 makes the arm track the targets, while the term in parenthesis pre-compensates for the motion of the mobile robot.

A useful simplification of this law is obtained by ignoring the pre-compensation term and by approximating the element  $\mathcal{J}_{2,3}$  by its first term, giving finally

$$\omega_\psi(t, Y) = k_4(Y_m - Y_{m\star}) / f. \quad (7)$$

This does not depend explicitly on our system's models, whose parameters may be imperfectly known, and will suffice to keep the targets in the field-of-view of the camera.

The control  $\mathcal{U}$  for the full system is then

$$\mathcal{U}(t, X, Y) = (v(t, X), \omega(t, X), \omega_\psi(t, X, Y))^\top. \quad (8)$$

A procedure for the estimation of the relative planar configuration of the target and the camera using three target feature points has been considered [18], similar to the one presented in Sugihara [17]. From the projections of the target features on the image plane, it is possible to estimate the angles with which these points are seen from the optical center  $C$ . Assuming that we know the target geometry, the relative configuration of the target with respect to the camera can be computed and the state  $(x, y, \theta)$  can be estimated using standard geometry.

However, since we are interested in positioning the mobile robot to the desired configuration  $X_\star = 0$ , while starting relatively close to it, we could attempt to do so without reconstructing explicitly its state. Since  $Y = \Phi(X)$ , the state  $X$  can be approximated up to first order by

$$\hat{X}(Y) = (J_\star)^{-1}(Y - Y_\star), \quad (9)$$

where  $J_\star \stackrel{\text{def}}{=} \frac{\partial \Phi}{\partial X}(0) = B_2(0) B_1(0)$ .

The proposed control law for the mobile manipulator can thus be expressed as a function of only the sensory data:

$$u = \mathcal{U}(t, Y). \quad (10)$$

### 3. Real-time Programming

Robotic applications are real time dynamical systems involving the combination of hardware, system software and dedicated algorithms. The successful design and implementation of an application strongly depends on the methodology used to organize these elements. The availability of CAD tools is a great help to the end-user to specify and to ensure the quality of the produced programs in terms of safety and performance. ORCCAD is a control architecture design environment (Simon, Espiau, Castillo and Kapellos [16], [3]) that we are currently developing to satisfy such requirements.

Systems where continuous and discrete event aspects interact are known as hybrid [1]. Robotic applications belong to this class of systems. Given that an established unified computational framework is not available to describe and analyze all the different aspects of hybrid systems, the ORCCAD system proposes a development methodology and a set of tools addressing coherently problems arising in all these aspects. Each step of the design of the application may involve the use of different classes of models: their interfacing is an important part of this procedure. This system makes some parts of this procedure transparent to the user by selecting and merging two computational models for the continuous and discrete-event parts of the application : the data-flow model for the specification of the former and the computation model defined by the semantics of the synchronous languages used for the specification of the latter. Their merging is obtained by the definition of two key entities: the *Robot-task* (RT), representing an elementary robotic action, where control aspects are predominant, encapsulated in a logical behavior expressed in terms of input/output events constituting its interface with the environment and the

other entities of the ORCCAD system; the *Robot-procedure* (RP), the basic element of which are the RTs, and where only behavioral aspects are considered. These aspects are specified, verified and implemented using the ESTEREL synchronous language whose semantics are expressed in terms of transition systems (automata). The above-mentioned events are characterized as pre-conditions, indicating that the RT can begin, exceptions, indicating critical situations inhibiting the execution of the RT and post-conditions, indicating the normal termination of the RT.

According to the proposed programming methodology, the first step towards the implementation of our experiment consists in identifying all the elementary actions needed to perform it: these correspond to the RTs. The basic idea is to associate a control law to a sub-objective of the experiment and to identify the set of events related to its execution. Subsequently, we specify and validate the identified RTs and we compose them in the form of a RP. After its validation, the whole specification is translated into an executable program to be loaded on the target architecture.

**Robot-task Specification and Validation :** Three sub-objectives, corresponding to three RTs, are identified for our application : the initialization, the stabilization of the mobile manipulator and the ending.

The INITIALIZATION RT drives the base and the arm in the working area and sets them in a desired given configuration. Two events are observed: i) a type-3 exception *joint\_lim* monitoring the joint limits of the arm and ii) a post-condition *pos\_reached* indicating that the pre-defined fixed configuration of the base and the arm have been reached.

The POINTSTABILIZATION RT starts after checking that the connection with the vision system has been established. During its execution the vision-based stabilization control law described in section 2 is executed and possible system errors, such as the camera losing sight of the target, are monitored. The control input  $U$  is given by equation 10. From the user point of view, the RT is specified by interconnecting a set of modules implementing algorithmic and behavioral functionalities. The internal structure of this RT is designed through the Graphical User Interface of ORCCAD and depicted in Figure 2a, where the following modules appear:

- READENC: reads the mobile manipulator state from wheel and arm joint encoders (wheel and joint position and velocity),
- VISION: reads the results of the image processing  $Y = (Y_l, Y_m, Y_r)^\top$ ,
- POSVISION: interprets the results of the image processing in order to compute  $\hat{X}(Y)$  given by equation 9 ,
- CONTROL: computes  $U(t, Y)$  given by equation 10.

The logical behavior of the RT is specified by the VIS\_PREC, POST and RTA modules as follows: i) the *pre-conditions* of the RT are that the connection with the vision machine is established and a valid information is obtained (*signal\_ok* event), ii) a type-2 *exception* is the loss of the video signal (*signal\_lost* event), iii) type-3 *exceptions*, like the arm reaching its joint limits and hardware failures, lead to an emergency stop of the vehicle (*joint\_lim*, *robot\_fail*, etc. events), iv) the *post-condition* of the RT is that the mobile robot reached its

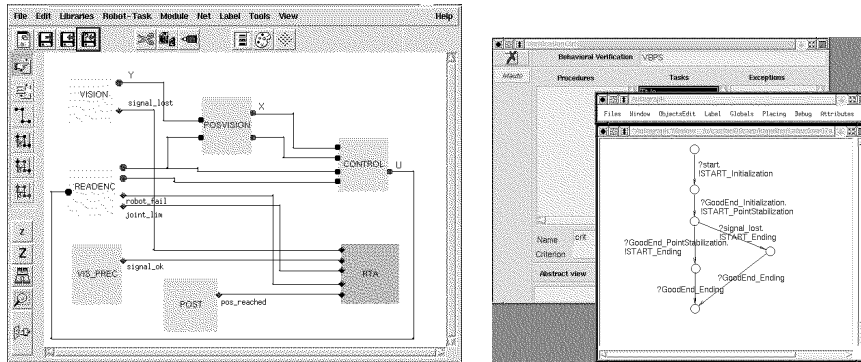


Figure 2. a) POINTSTABILIZATION RT b) VBPS automaton observation

final position (*pos\_reached* event).

The specification is completed by assigning a sampling period of 40 msec (video rate) to all modules. In addition, non-blocking message passing mechanisms are selected for inter-module communication.

The ENDING RT drives the arm to its home position in a similar way as for the INITIALIZATION RT.

**Robot-procedure Synthesis :** The VBPS RP is designed to specify the evolution of the application as the composition of the previously defined RTs. It simply states that the application will start after user confirmation, which corresponds to the pre-condition *start*; its nominal execution consists of the sequencing of the three RTs: INITIALIZATION, POINTSTABILIZATION and ENDING:

```

RP VBPS
[
pre-condition:      start
<main_program>
                    INITIALIZATION();
                    POINTSTABILIZATION();
                    ENDING();
exception_type2:   (signal_lost),
                    ENDING();
exception_type3:   (robot_fail), (sensor_fail), (joint_lim), ...
]

```

This specification of the RP is programmed in ESTEREL. Its validation can be done in the ORCCAD environment. First, the safety property (any fatal exception is appropriately handled by emission of a specific signal leaving the system in a safe situation) and the liveness property (the RP always performs its goal in a nominal execution) are automatically proven.

Then, the conformity of the RP behavior with mission constraints can be verified, by observing the global automaton with respect to criteria restricted only to events *relevant* to these constraints and indicated by the user in terms of RTs and exceptions. Figure 2b illustrates this process, for the case when the user indicates that he is interested to see the relation between the RTs and the

type-2 exception *signal\_lost*. The resulting automaton, after observation, is shown: at the occurrence of a *signal\_lost* event, the ENDING action is required to start.

**Real-Time Robot Control Software :** ORCCAD provides tools for the specification and formal verification of robotic applications as well as the automatic generation of the corresponding real-time code [10], integrated within a set of dedicated graphical user interfaces, several aspects of which are illustrated below. The whole robot control software is obtained by the translation of the RP specification into C++ code, independent of the real-time target environment (operating system, computational hardware), which is subsequently integrated with its run-time libraries. It is made of three parts:

- a set of real-time tasks, usually periodic and synchronized according to consumer/producer protocols, dedicated to the computation of the continuous aspects of the system (mainly the control laws);
- the discrete event automaton directly generated in C by the ESTEREL compiler;
- the interface between the two previous parts. Its role is dual: it transforms significant changes occurring in the continuous aspects of the system into the discrete events that are fed to the automaton. And vice versa, it links each output event produced by the automaton with appropriate requests made to the real-time task layer.

**Robot Control Software Generation :** From a graphical and textual high-level specification of the robotic application, the ORCCAD environment automatically produces the corresponding real-time program to download and execute on the target architecture. The real-time operating system used is VxWORKS 5.2. The generated code runs on a single processor at video rate.

## 4. Experimental Results

**The Robotic System :** Our test-bed is a unicycle-type mobile robot carrying a 6 d.o.f. manipulator arm with a CCD camera and equipped with a belt of eight ultrasonic sounders (see figure 3) ([12], [13]). The on-board computer architecture is built around a VME backplane. The robot controller is based on a Motorola MVME 162 with IP modules ensuring an actuator velocity servo-loop at a rate of 1ms.

**Vision Hardware and Software :** To overcome difficulties in image processing, induced by strong real-time constraints and the processing of large amounts of data, we have developed a vision machine [12] characterized by its modularity and its real-time capabilities. Its architecture is based on VLSI chips for low level processing and on DSP processors for more elaborate processing. To facilitate program development, the vision machine has been implemented on an independent VME rack outside of the mobile robot, keeping however the possibility to transfer the vision boards to the on-board rack. The software used on the vision machine implements the concept of active window [12]. An active window is attached to a particular region of interest in the image and its task is to extract a desired feature in this region. At each active

window we associate a Kalman filter able to perform the tracking of the feature along the image sequence. Several active windows can be defined, at the same time, in the image and different types of processing can be done in each window. In our application, the visual data ( $Y_l$ ,  $Y_m$ ,  $Y_r$ ) are extracted at video rate from 3 active windows, which track the targets as the camera moves (see figure 4).

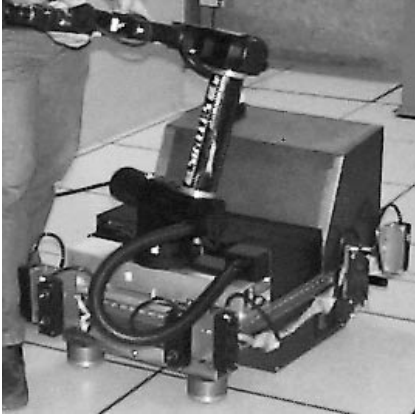


Figure 3 : Mobile Manipulator

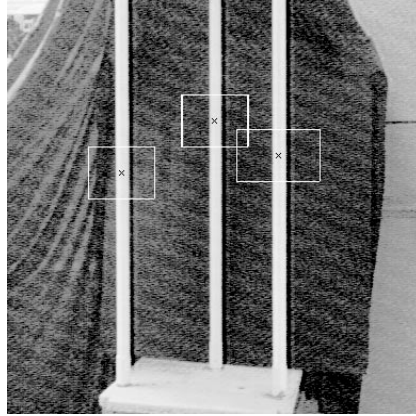


Figure 4 : Active windows

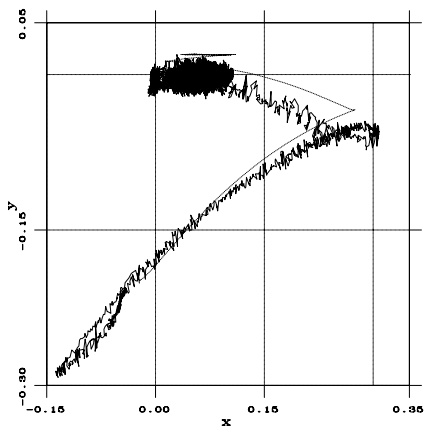
**Experiments :** In the experimental results presented below, we use the control law 10 with the unicycle controls 4, the arm control 7 and the state approximation 9 by sensory data. The following parameters corresponding to the models of section 2 are used:  $l_1 = 0.51 \text{ m}$ ,  $l_2 = 0.11 \text{ m}$ ,  $d = 2.95 \text{ m}$ ,  $f = 1 \text{ m}$ . The following gains are used for the above control laws:  $w = 0.1$ ,  $k_1 = 0.25$ ,  $k_2 = 2$ ,  $k_3 = 100$ ,  $\alpha = 10$ ,  $k_4 = 12$ .

The visual data ( $Y_l$ ,  $Y_m$ ,  $Y_r$ ) are acquired and pre-processed by the vision system, which transmits them to the robot controller for use in implementing the above control laws. This process introduces a delay  $\delta$  to the system, which, in our case, equals to 3 sampling periods (120 msec). Thus, the visual data used to control the system at time  $t$  correspond to its state at time  $t - \delta$ . This degrades the performance of the control and cannot be corrected by e.g. increasing the gains of the control laws. To compensate for these delays, we use the arm joint and robot wheel encoders to calculate corrections to be added to the controls for the arm and for the wheel angles. In the case of e.g. the arm, this compensation takes the form of the term  $-k_4[\psi(t) - \psi(t - \delta)]$ , which is added to the control law 7 and where  $\psi(t)$  is the arm encoder reading at time  $t$ . A comparison of the step responses of the arm, in the case where delay compensation is used and where no such compensation is used, shows that the long oscillations that the uncompensated arm does before settling down can be significantly reduced.

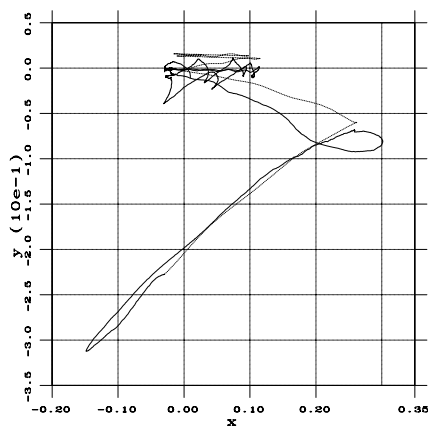
Initial experiments used the raw visual data to calculate the state and the controls. The resulting  $(x, y)$ -trajectory is plotted in figure 5. In the following figures, the dotted lines represent data obtained by odometry, while solid ones are data obtained by vision. As it is evident from this figure, implementation of

this scheme leads to significant small oscillations and jerks during the motion of the system.

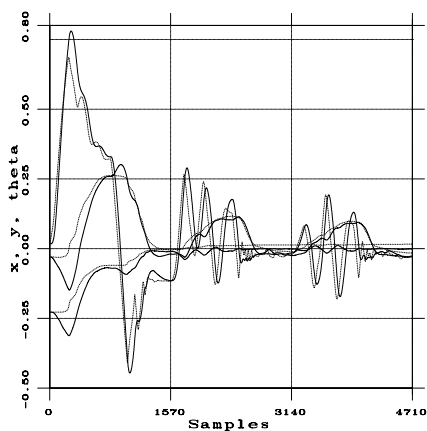
Subsequent experiments used Kalman filtering of each of the state variables  $(x, y, \theta)$ , to make the corresponding trajectories smoother and compensation of the vision-induced delays was introduced. No filtering was used on the visual data themselves. The resulting  $(x, y)$ -trajectory, the evolution of the state  $(x, y, \theta)$ , of the unfiltered visual data  $Y_m$ , as well as the corresponding controls  $v, \omega$  are shown in figures 6–10 when the system starts at  $(x, y, \theta) = (-0.03, -0.23, 0.02)$ . Each period of the time-varying controls corresponds to 1570 samples (data on the state of the system are recorded every 40 msec). The system motion is noticeably ameliorated.



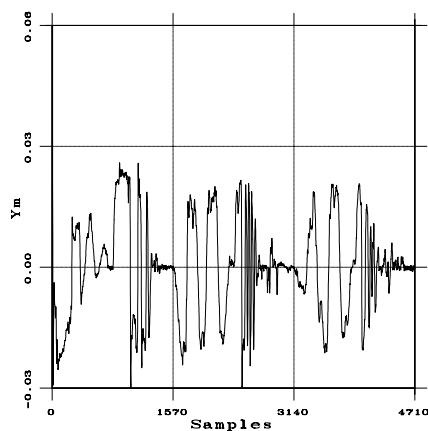
$(x, y)$ -trajectory: No state filtering  
Figure 5



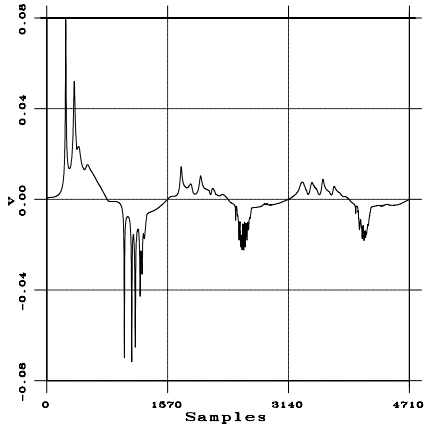
$(x, y)$ -trajectory: Filtered state  
Figure 6



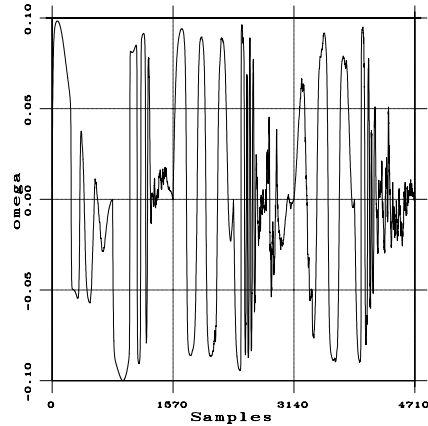
$x, y, \theta$  : Filtered state  
Figure 7



$Y_m$  : Visual data (Unfiltered)  
Figure 8



$v$  : Time-varying heading speed  
Figure 9



$\omega$  : Time-varying angular velocity  
Figure 10

The results presented here are preliminary, demonstrating the feasibility of the proposed method, leaving however several issues to be further studied, such as the difference between the  $(x, y)$ -trajectories obtained by vision and odometry in figure 6, possibly due to calibration and state approximation errors, and the excitation of small oscillations near the desired final configuration (c.f. figures 7–10), possibly due to robustness problems of the controls in the presence of model parameter errors.

## 5. Conclusions

The feasibility of a novel class of vision-based point stabilization techniques for nonholonomic mobile manipulators was demonstrated in this work. Their robustness with respect to noisy data, delays and modeling errors is currently under study. We consider the planar case, where a one-d.o.f. arm carrying a camera moves parallel to the plane supporting the mobile base. The case of an arm with more d.o.f. and a more general visual setting is also currently under investigation.

## References

- [1] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis and S. Yovine, “The Algorithmic Analysis of Hybrid Systems”, *Theoretical Computer Science*, Vol. 137, 1995.
- [2] B. Espiau, F. Chaumette and P. Rives, “A New Approach to Visual Servoing in Robotics”, *IEEE Trans. on Robotics and Automation* **8**, 313-326, 1992.
- [3] B. Espiau, K. Kapellos and M. Jourdan, “Verification in Robotics: Why and How?”, *Robotics Research*, The 7th Intl. Symposium, Eds. G. Giralt and G. Hirzinger, Springer Verlag, 1995.
- [4] G.D. Hager and S. Hutchinson, Eds., “Vision-based Control of Robotic Manipulators”, Special section of *IEEE Trans. Robotics and Automation* **12**, 649-774, 1996.

- [5] K. Hashimoto, Ed., *Visual Servoing*, World Scientific, 1993.
- [6] R.T. M'Closkey and R.M. Murray, "Nonholonomic Systems and Exponential Convergence: Some Analysis Tools", *IEEE Conf. on Decision and Control*, San Antonio, Texas, 1993.
- [7] R.T. M'Closkey and R.M. Murray, "Exponential Stabilization of Driftless Non-linear Control Systems Using Homogeneous Feedback", preprint, Caltech, 1995.
- [8] P. Morin and C. Samson, "Application of Backstepping Techniques to the Time-Varying Exponential Stabilization of Chained Form Systems", INRIA Research Report No. 2792, Sophia-Antipolis, 1996.
- [9] R. Pissard-Gibollet and P. Rives, "Applying Visual Servoing Techniques to Control a Mobile Hand-Eye System", *IEEE Intl. Conf. on Robotics and Automation*, 1995.
- [10] R. Pissard-Gibollet, K. Kapellos, P. Rives and J.J. Borrelly, "Real-Time Programming of Mobile Robot Actions Using Advanced Control Techniques", *Experimental Robotics IV*, The 4th International Symposium, Eds. O. Khatib and J.K. Salisbury, pp. 565-574, Stanford, USA, June 30-July 2, 1995.
- [11] J.-B. Pomet and C. Samson, "Time-Varying Exponential Stabilization of Non-holonomic Systems in Power Form", INRIA Research Report No. 2126, Sophia-Antipolis, 1993.
- [12] P. Rives, J.J. Borrelly, J. Gallice and P. Martinet, "A Versatile Parallel Architecture for Vision Based Applications", *Workshop on Computer Architecture for Machine Perception*, New Orleans, 1993.
- [13] P. Rives, R. Pissard-Gibollet and K. Kapellos, "Development of a Reactive Mobile Robot Using Real-Time Vision", Third International Symposium on Experimental Robotics, Kyoto, Japan, October 28-30, 1993.
- [14] C. Samson, "Velocity and Torque Feedback Control of a Nonholonomic Cart", in *Advanced Robot Control*, Ed. C. Canudas de Wit, Lecture Notes in Control and Information Sciences, No. 162, Springer-Verlag, 1990.
- [15] C. Samson, "Time-varying Feedback Stabilization of Car-like Wheeled Mobile Robots", *The International Journal of Robotics Research* **12**, 55-64, 1993.
- [16] D. Simon, B. Espiau, E. Castillo and K. Kapellos, "Computer-aided design of generic robot controller handling reactivity and real-time control issues", *IEEE Trans. on Control Systems and Technology* **1**, 1-24, 1993.
- [17] K. Sugihara, "Some Location Problems for Robot Navigation Using a Single Camera", *Computer Vision, Graphics and Image Processing* **42**, 112-129, 1988.
- [18] D.P. Tsakiris, C. Samson and P. Rives, "Vision-based Time-varying Mobile Robot Control", Final European Robotics Network (ERNET) Workshop, Darmstadt, Germany, September 9-10, 1996. Published in *Advances in Robotics: The ERNET Perspective*, Eds. C. Bonivento, C. Melchiorri and H. Tolle, pp. 163-172, World Scientific Publishing Co., 1996.
- [19] D.P. Tsakiris, C. Samson and P. Rives, "Vision-based Time-varying Stabilization of a Mobile Manipulator", Proceedings of the Fourth International Conference on Control, Automation, Robotics and Vision (*ICARCV'96*), pp. 2212-2216, Westin Stamford, Singapore, December 4-6, 1996 (Available on WWW in URL: <http://www.inria.fr/icare/personnel/tsakiris>).