Preprints of the Third International
Symposium on Experimental Robotics
Kyoto, Japan, Oct. 28-30, 1993

Development of a Reactive Mobile Robot Using Real Time Vision

Patrick Rives* , Roger Pissard-Gibollet**, Konstantinos Kapellos**

* INRIA Sophia Antipolis
2004 Route des Lucioles
06565 Valbonne Cedex - France
rives@sophia.inria.fr

** ISIA-Ecole des Mines de Paris
Rue Claude Daunesse
06565 Valbonne Cedex - France
pissard(kapellos)@isia.cma.fr

Abstract

In this paper, we discuss both theoretical and implementation issues of a vision based control approach applied to a mobile robot. After having briefly presented a visual servoing framework based on the task function approach, we point out some problems of its application to the case of mobile nonholonomic robots. We will show how using additional degrees of freedom provided by a manipulator arm, allows to overpass these difficulties by introducing redundancy with respect to the task. The second part of the paper deals with the development of an experimental testbed specially designed to rapidly implement and validate reactive vision based tasks. It is constituted by a mobile robot carrying a hand-eye system using a dedicated vision hardware based on VLSI's and DSP 96002's processors, which allows to perform image processing at video rate. Programming aspects are fully taken into account from tasks level specification up to real time implementation by means of powerfull software tools handling explicitely reactivity and real time control issues.

1. Introduction

Recent improvements in the field of vision sensors and image processing allow to hope that the use of vision data directly into the control loop of a robot is no more an utopic solution. Commonly, the general approach in robot vision is the following: processing vision data into the frame linked to the sensor, converting data into the frame linked to the scene by means of inverse calibration matrix and computing the control vector of the robot into the frame linked to the scene. This scheme works in open loop with respect to vision data and cannot take into account errors in the measurement or in the estimation of the robot position. Such an approach needs to perfectly overcome the constraints of the problem: the geometry of the sensor (for example, in a stereovision method), the model of the environment and the model of the robot. In some cases, this approach is the only one possible but, in many cases, an alternative way consists in specifying the problem in terms of control directly into the sensor frame. This approach is often referred as visual servoing [WEI 84] or sensor based control [ESP 87]; in this case, a closed loop can be really performed with respect to the environment and allows to compensate the perturbations by a robust control scheme. Moreover, this approach seems to be a fruitful way to implement robotics tasks based on *reactivity concept* [BRO 86]. The work described in this paper deals with such an approach applied to autonomous mobile robots using vision sensor. More particularly, we will focus on the development of our experimental testbed including a mobile robot carrying a hand-eye system and the dedicated vision hardware which has been developped for testing reactive approaches based on visual servoing.

[0]The work on mobile robot is supported by firm ECA (Toulon, France) under a CIFRE contract. The work on Orccad system is made in cooperation with Aleph Technologies (Grenoble, France).

Part I
Theoretical Aspects

2. Sensor Based Tasks Framework

As a preliminary, it must be noted that the framework which will be presented here, only addresses local level task specification aspects using sensor based control approaches. Therefore, it excludes global level task specification aspects like mission planning or off-line programming. At this level of specification, a robotics application will be stated as a concatenation of elementary sensor based tasks. At each elementary task is associated a servoing law, continuous on a time interval $[T_{begin}, T_{end}]$, and built from the sensor's outputs. The transitions between the elementary tasks are scheduled by the events occuring during the task (reactive approach).

2.1. Elementary Sensor based Task Specification:

At this level, we want to provide a formalism to easily specify servoing tasks controlling the local interactions between the robot and its environment. A natural way is to specify these interactions in terms of kinematics constraints between a frame linked to the robot and a certain frame associated to a part of the local environment. By analogy with the *Theory of Mechanism*, we call this set of constraints : **a linkage**. The class of the linkage is so defined by the dimension of the subspace of compatible velocities which are let unconstrained by the linkage. This subspace of compatible velocities can be also viewed as a screw space and it fully characterizes the interactions between the robot and its environment. This concept of linkage plays an essential role in the synthesis of control laws using contact sensors for assembly applications. It can be easily extended to the case of no contact sensors like proximity or vision sensors. So, we can define the notion of *virtual linkage* by replacing the mechanical constraints by, for example, optical constraints like bringing the optical axis of a camera into alignment with a certain axis of a frame attached to the scene. After specifying an elementary task in terms of a peculiar virtual linkage to be reached between the frame attached to the robot and the frame attached to the environment, we need to realize this virtual linkage by using sensors data. To do that, it is necessary to identify the mapping between the configuration space, characterizing the relative position of the robot with respect to its local environment, and the sensor response space.

2.2. Modeling the Interactions between the Sensor and its local environment

We only give the main characteristics of this approach, more inquires can be found in [SAM 90], [CHA 90] and [ESP 92]. A sensor (S), providing a multidimensional signal $\underline{s}$ is linked to a robot (R) with related frame F_S. (S) interacts with an environment at which is associated a frame F_T. The position (location and attitude) of (R) is an element $\bar{r}$ of SE_3 (Special Euclidean Group) which is a six dimensional differential manifold. Let us assume that the output of (S) is fully characterized by the relative position of (S) with respect to (T) (for example, if we use vision sensor, we don't consider image changes dues to lightning variations). Using this assumption, the sensor's output is modeled as a function $s(\bar{r}, t)$ which is fully defined by the nature of the sensor, the position of the robot $\bar{r}$ and the time t representing the own motion of the part of environment associated to the frame F_T. This function represents a mapping between the configuration space of the robot and the sensor response space. For many types of sensors (proximity, vision, force...), an analytical form of this function can be found. Due to the properties of the screw spaces (see [ESP 92] for more details) the differential form of $s(\bar{r}, t)$, can be expressed as a screw product :

$$\dot{\underline{s}} = H \bullet T_{CT}, \tag{1}$$

where
H is called the *interaction screw*, T_{CT} is the velocity screw characterizing the relative motion between F_S and F_T and $\bullet$ represents the screw product.

Considering the bilinear form L^T of the interaction screw H expressed in the sensor frame F_C, and with an obvious breach of notation, we obtain :

$$\dot{\underline{s}} = L^T . T_{CT}, \tag{2}$$

Interaction matrix may be derived for many exteroceptive sensors. In the case of vision sensor, the vector $\underline{s}$ may be constituted from the projection of geometrical primitives in the environment like points, lines, circle... An analytical expression for the interaction matrix when the image features are general algebraic curves can be also obtained (for more details, see [CHA 90]). As a main result, we can note that the interaction screw formalism is a powerful means to model the sensor behavior. It relates both the value of the signal to the relative position between the sensor and the target and its variation to their relative motions. Moreover, due to the properties of the screw space, it can be easily expressed in terms of virtual linkage.

2.3. Relating Interaction Matrix and Virtual Linkage

Let us come back to the specification of a robotics application as a concatenation of elementary subtasks expressed in terms of a succession of virtual linkages to be realized between the robot and its environment. With no lack of generality, we now focus on the use of vision sensors. Let us consider the case where visual signal output is invariant ($\underline{\dot{s}} = 0$) with respect to a set of camera motions defined by T^*_{CT} :

$$\underline{\dot{s}} = L^T.T^*_{CT} = 0 \tag{3}$$

It can be easily seen that :

$$T^*_{CT} = Ker(L^T) \tag{4}$$

where $Ker(L^T)$ is the kernel of interaction matrix.

Now, recalling the definition of the virtual linkage, we find out that the subspace spanned by T^*_{CT} is precisely the subspace of compatible velocities which leads no change in the image during the motion of the camera. According to this fact, it becomes possible to define a virtual linkage by a set of elementary visual signals in the image invariant with respect to a motion belonging to the subspace of compatible velocities spanned by the virtual linkage. To perform a specific virtual linkage, many different features in the image can be used like points, lines, area, inertial moments and so on... In practice, their choice is guided by the application : a priori knowledge about the scene, low cost image processing, robustness with respect to the measurement noise...

By this formalism, this approach gives a fruitful way for specifying an elementary vision based task in terms of an *image target* to be reached when the task is well performed. We think it is a straightforward manner to implement reactivity approaches using vision sensors. In the next section, we will show how such an approach can be embedded in a robust closed loop control scheme using the *task function formalism* [SAM 90].

2.4. Sensor Based Task Function

A now classic approach in robotics it to consider the process of achieving a task such as tracking or positioning as a problem of regulation to zero of a certain output function: *the task function.* For example, a trajectory following task in cartesian space can be expressed as a regulation to zero of the task function: $e(t) = \bar{r}(t) - R^*(t)$, where R^* is the desired trajectory expressed in SE_3. We consider that the task is perfectly achieved if $\|e(t)\| = 0$ during the time interval $[T_{begin}, T_{end}]$ (see [SAM 90] for full details about the task function approach in robot control). The application of the task function approach to vision based tasks is straightforward: the task is now described as a desired values for a certain set of elementary visual signals to obtain. In mathematical terms the general definition of a visual task function vector e is :

$$e\ (t) = C\ (s\ (\bar{r}, t)) - s^*(t)) \tag{5}$$

where

- s^* can be considered as a desired value of the vector of the elementary visual signals (reference image target) to be reached in the image.
- $s\ (\bar{r}, t)$ is the value of the visual features currently observed by the camera, which only depends on the position between the camera and the scene.
- C is a matrix which allows, for robustness issues, to take into account more visual features than necessary. This matrix must have some properties as we will see later, for ensuring convergency at control level.

The quantity which arises naturally in our output regulation problem is the *task error* $\dot{e}$. For simplicity, we deal with a very simple example of *velocity control scheme* using visual feedback (for more advanced control schemes, see [SAM 90]). Assuming the presence of an appropriate robot's velocity servoing loop in Cartesian space , we can use as control input, the velocity T_c of the camera. We thus may choose the following control law :

$$T_c = -\lambda e \tag{6}$$

with $\lambda > 0$.

Let us consider the assumption that e is only depending of $(\bar{r}, t)$, we get :

$$\dot{e} = \frac{\partial e}{\partial s}.\frac{\partial s}{\partial \bar{r}}.T_c + \frac{\partial e}{\partial t} \tag{7}$$

$$\dot{e} = C.L^T.T_c + \frac{\partial e}{\partial t} \tag{8}$$

where $\frac{\partial e}{\partial t}$ parametrizes an eventual motion of the target. In the case of a motionless target $\frac{\partial e}{\partial t} = 0$, and we obtain :

$$\dot{e} = C.L^T.T_c = -\lambda\ C.L^T.e \tag{9}$$

An exponential convergence will thus be ensured under the sufficient condition:

$$CL^T > 0 \tag{10}$$

in the sense that a $n \times n$ matrix A is positive if $x^T Ax > 0$ for any nonzero $x \in \mathbf{R}^n$.

A good and simple way to satisfy this convergence condition in the neighbourhood of the desired position is to choose for the matrix C the generalized inverse of the interaction matrix associated to s^*:

$$C = L^{T^+}{}_{|s=s^*} \tag{11}$$

2.5. Vision Based Control Applied to a Mobile Robot

Applying vision based control theory to mobile robots provides an interesting framework for implementing reactive applications. However it is necessary to take some care due to the presence of nonholonomic constraints. Because of this, it is now wellknown [SAM 91] that pure state feedback stabilization of a two-wheels driven nonholonomic cart's configuration around a given terminal configuration is not possible, despite the controllability of the system.

As a consequence, we cannot ensure a convergence of the task error by using simple gradient-like methods in the image frame, for a camera rigidly attached to the mobile base. One way to avoid this problem consists in adding some degrees of freedom to the camera by mounting it on the wrist of a manipulator. Then, if we consider the whole mechanical system constituted by the cart and its manipulator like a single kinematic chain, we can show that the mapping between the configuration space (d.o.f of the cart + d.o.f of the manipulator) and the position (attitude and location) of the camera is almost everywhere regular (excepted in some isolated configuration) [RIV 92] and it becomes possible to fully control the motion of the camera without being limited by the cart nonholonomic constraints. By this way, we can execute visual servoing tasks directly defined in the camera frame in the same manner that in the case of holonomic systems. Obviously, in such an approach, we will not be able to control explicitely the trajectory of the cart but, by using some trivial geometric considerations, we can bound the volume which will be occupied by the cart during its displacement. We apply this approach to our system constituted by a two wheels driven cart with a camera mounted on a two d.o.f manipulator. The figure 1 shows simulation results of such an approach where the vision based task consists of wall following by servoing on the two parallel lines corresponding to the skirting boards at the bottom of the walls.

3. ORCCAD Principles

As previously seen, elementary servoing tasks, such as visual servoing, can be efficiently solved in real-time using closed loops.

On the other hand, to execute an application, considered like a temporal arrangement of elementary servo-control tasks, the robotic system must be able to react to occurences of various events. Events from the environment may trigger or kill elementary servo-control tasks or activate a particular recovering proccess. Also, events generated by the robotic system itself, informing for the normal or failed termination of

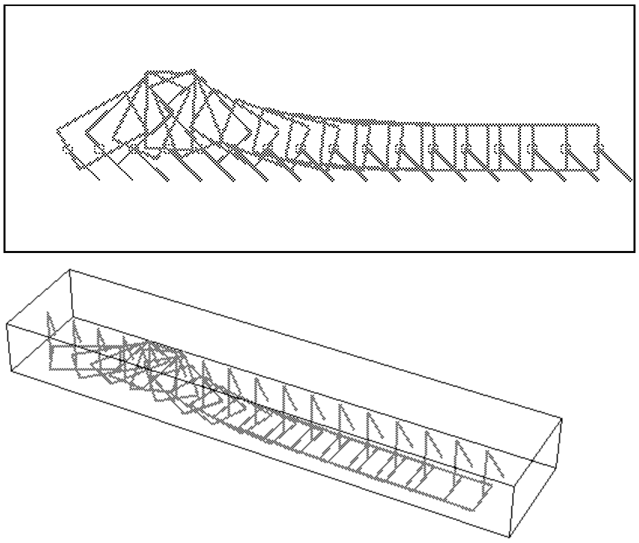

Figure 1. Wall following

elementary servoing tasks, must be as well considered.

Therefore a rigourous methodology is required to design such a robotic process. It must handle continuous and discret time aspects. A reliable robot programming environment must be provided. These requirements are at the basis of the development of the ORCCAD system [SIM 93], a generic robot controller handling reactivity and real-time issues.

We give a short overview of the ORCCAD's general concepts and refer the reader to [SIM 93] and [COS 92] for a detailed presentation. We will mainly focus our description on its key entity that models servo-control tasks and their associated event-based behaviour, the so called Robot-task.

3.1. Robot-task

A Robot-task is formally defined as the parametrized entire specification of:

- an elementary servo-control task, i.e the activation of a control scheme structurally invariant along the task duration
- a logical behavior associated with a set of signals liable to occur previously to and during the task execution.

Several stages exist in the design of a Robot-task. Firstly, the elementary servo-control task should be defined: in a first step, the formal specification in continuous time which characterizes the task from the automatic control point of view has to be established.

Then, it has to be extended to take into account implementation issues: discretization, variable quantization, delays, computation times, periods .... Finally, starting, stopping, killing of the task and controlling its evolution require to define adequate signals and to build an automaton handling the overall behavior. This design process is presented later through an example.

The Event-based Behavior In a way, a Robot-task is atomic for the application level. However it follows an internal sequencing which has not to be seen in normal (failure-free) circumstances. Nevertheless the Robot-task has also to exchange information with other Robot-tasks, in order to synchronize and/or condition their activation. In the ORCCAD framework these two aspects are considered in a single way. The used approach is the one of reactive systems with the synchrony assumption [BEN 91]: signals are emitted without any absolute reference time from and to an automaton which specifies the Robot-task behavior.

In ORCCAD, all signals and associated processings belong to well defined categories. The following types are distinguished:

- the pre-conditions: Their occurence is required for starting the servoing task. They may be pure synchronization signals or signals related to the environment usually obtained by a sensor.
- the exceptions: are generated during the execution of the servoing task and indicates a failure detection. The exception processing is structured as follows:
  - type 1 processing: the reaction to the received exception is limited to the modification of the value of at least one parameter of the servoing control (gain tuning, for example).
  - type 2 processing: the exception requires the activation of a new Robot-task. The current one is therefore killed. When the ending is correct, the nominal post-conditions are fulfilled. Otherwise, a specific signal is emitted to the application level, which knows the recovering process to activate.
  - type 3 processing: the exception is considered as fatal. Then everything is stopped.
- the post-conditions: They are either logical synchronization signals emitted by the Robot-task automaton itself in case of correct termination, or signals related to the environment handled as conditions for a normal termination of the Robot-task.

Part II
Implementation Aspects

4. Description of the Robotic System

We have developped a highly versatile testbed in order to validate reactive vision based approaches in real experiments. This robotic system uses a cart-like mobile robot carrying a two d.o.f head with a CCD camera. The typical application domain concerns automatic exploration of an indoor environment. So we have designed a mobile robot light enough (75 Kg), easy to handle (cart-like), and with reduced size (lenght: 700 mm, width: 500 mm, height: 600 mm).

The on board computer architecture is built around a VME backplane which can accept until 12 VME boards. For these experiments we use a Motorola MVME 167 and a 6 axes custom-made servocontrol board based on dedicated Hewlett Packard components (HCTL 1100) and allowing to control different type of motors (DC, DC brushless, step motor).

The mobile robot disposes of two switchable power supplies made of an extern supply and two on board batteries. During the development step, an umbilical link is used both for power and communication (ethernet thin link, three video channels and two serial links). The Robot-task is designed and checked on a Sun Sparc workstation; then it is downloaded to the robot through ethernet. During the execution, the monitoring of the robot can also be done through ethernet. Figure 2 presents an overview of the whole experimental testbed.

Implementing efficiently a vision based control approach needs to solve several problems at the image processing level. The major difficulties are induced by strong real time constraints and the processing of large dataflow from the sequence of images. To overpass this difficulties, we have developped an original machine vision ([MAR 91], [RIV 93]) characterized by its modularity and its real time capabilities (figure 3). Its architecture is based both on VLSI chips for low level processing and on multi DSP's processors for more elaborated processings. It implements the concept of active window. An active window is attached to a particular region of interest in the image and has in charge to extract a desired feature in this region of interest.

At each active window is associated a temporal filter able to perform the tracking of the feature along the sequence. Several active windows can be defined, at the same time, in the image and different processings can be done in each window. For facilities of development, this machine vision has been implemented in an independant VME rack outside of the robot. However, for experiments which require high autonomy, these boards can be plugged in the on board rack.

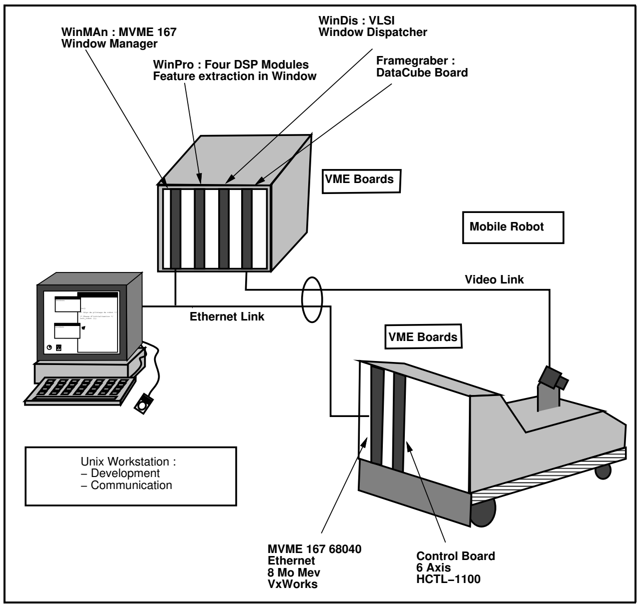

Figure 2. robotic system

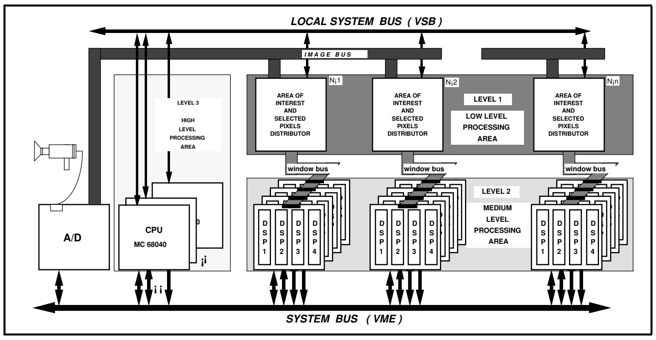

Figure 3. Overview of the machine vision

5. Visual Servoing Robot-task Design

Such a robotic system is able to support applications constitued by a sequence of visual servoing Robot-tasks. So we need, the strong formalisms and the efficient design methodology described above, for implementing general control laws and taking in account robot or environment events.

The ORCCAD system provides a set of CAD tools allowing to design, check and implement Robot-tasks. An object-oriented approach has been choosen for modeling the Robot-task. A given Robot-task is then fully specified by instantiation of required objects leaves of a set of class hierarchies structuring a robotic system, i.e four classes related to the control scheme, two classes related to the physical entities involved by the system, two classes related to the logical (event-based) behavior of the robot task (observers and automaton of the Robot-task). These objects are defined by the functional and the temporal set of attributes and methods concerning coherence testing, automatic code generation for simulation and execution, ...

This modelling is at the basis of an HMI allowing the control expert to specify a Robot-task. Let us now describe the different stages of a visual servoing Robot-task design.

Continuous time specification At this stage of design the end-user's specification, done in natural language, provides information about actions to be performed, information about the robotic system to be used, events to be considered. Here, the end's user task specification could be *the robot goes in front of the door* or *the robot follows wall.*

For a control expert, a first stage, is to see if the problem is well conditionned and to choose an adequate control law. In our case, we will use a visual servoing technique to complete the task in a robust way with respect to robot's environment changes.

The continuous-time control law achieving the specified action is specified in term of block-diagrams and/or equations. For a two-wheeled mobile robot driven by visual servoing Robot-task, the control law computes the motor velocities (wheels and head axis) to be applied from an error between the desired and the current value of the visual feature to be reached. Usually, such a scheme may be splitted in several functionnal modules which exchange data. In figure (4) we could see the functionnal decomposition where :

- *GT_VS* generates a visual features trajectory in the image space to be followed : $s^*(t)$.
- *WINFEAT* interfaces the Robot-task with the real time vision system. It furnishes the current visual features parameters $s(\bar{r}, t)$.
- *ERROR_VS* computes the error signal in the image space: $\epsilon = s(\bar{r}, t) - s^*(t)$.
- *TF_VS* computes the control law using the task function approach : $e = C\epsilon$ and $T_c = -\lambda e$.

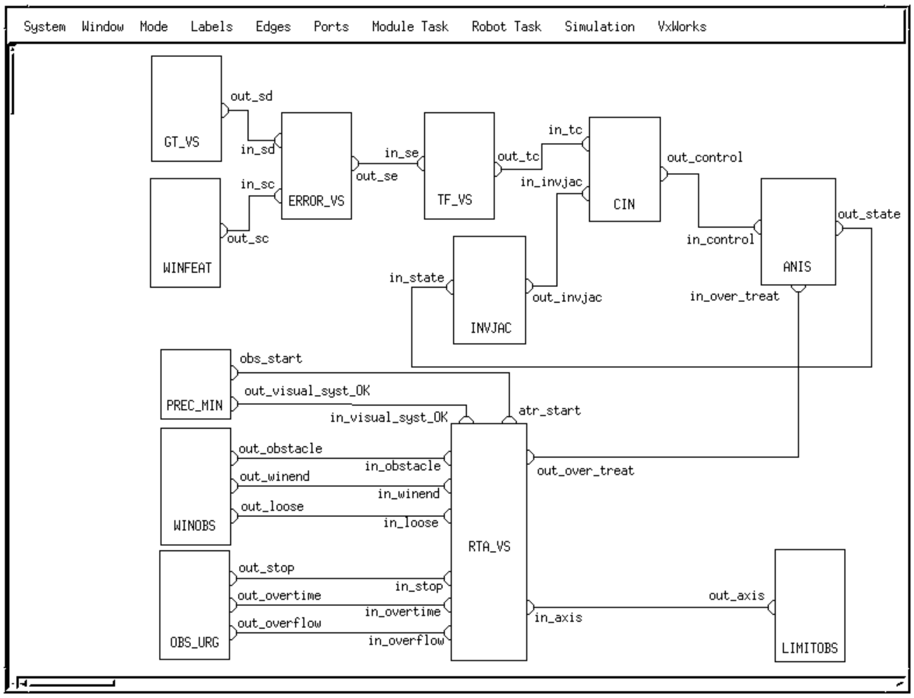

Figure 4. visual servoing Robot-task

- *INVJAC* computes the jacobian robot matrix J_R^{-1} ([RIV 92]).
- *CIN* sends control-inputs to the robot's interface : $\dot{q} = J_R^{-1} T_c$.
- *ANIS* interfaces or simulates the whole mechanical system (mobile robot and its head) to be controlled.

At this stage of the control law conception the Robot-task designer chooses the classes to be instantiated and defines the objects (so called F-objects) considering the functional set of attributs (minimum values for closed loop gains, initial and final positions, location of sensors ...). The ORCCAD system guides, as far as possible, the user's choices and checks the specifications coherence.

The result of this stage of design is the specification of the functional graph of the Robot-task which is an oriented graph, the nodes of which are the F-objects instantiated for the Robot-task, and the edges of which correspond to the existence of formal data transfers between F-objects.

Time constrained specification The passage from the above continuous-time specification to a description taking into account implementation aspects is done by specifying at this level the *temporal properties*, i.e sampling periods, durations of computations,

```
Initialization code
Every (T-period){
Reading all input ports
Data processing code
Writing all output ports}
```

Figure 5. Module-task structure

communication and synchronization between the processes, data processing code, ...

At this stage of specification the basic entity manipulated is the Module-task. It is a real-time task used to implement an elementary functional module of the control law. It owns temporal properties, like its computation duration and its activation period. Since the Module-tasks may, possibly, be distributed over a multiprocessor target architecture they communicate using typed ports (8 communication and synchronization mechanisms are provided). Each Module-task owns one input port for each data it handles and one output port for each data it produces. The structure of the Module-tasks separates calculations, related to control algorithms issues, and communications, related to implementation issues and operating system calls (see fig 5 illustrating the structure of a periodic Module-task).

Since I/O ports are the only visible parts of the Module-tasks and their connections are given by the designer of the Robot-task, the possibility to reuse the Module-tasks as objects in different Robot-task schemes exists with no need of internal data structure modification.

At this step the Robot-task designer adds to the already created F-objects properties characterizing ports (name, synchronization type, name and type of exchanged variables), localization of associated codes (initialization, computation), period and duration of the Module-taskse and name of the dedicated processor creating the MT-objects.

In the visual servoing task, all communication types are *asynchronous/asynchronous* excepted between the *CIN* and *ANIS* Module-task where the synchronisation is *asynchronous/synchronous*. Sampling period for visual Module-tasks (*GT_VS, WINFEAT, ERROR_VS, TF_VS*) is lower (40 ms) than for robot Module-tasks (*INJ_JAC, CIN*) which is 10 ms. The robot interface Module-task *ANIS* is not periodic but it is synchronised by the *CIN* Module-task.

The result of this stage of design is the time-constrained specification graph (see figure 4) of the Robot-task which is an oriented graph, the nodes of which are the MT-objects instantiated for the Robot-task, each arrow corresponds to the transfer of a variable from a Module-task to another one.

The Event-based Behavior of the Robot-task The servoing visual Robot-task behavior is driven by nine events which allow to monitor the task :

- two pre-condition events (monitoring by the observer *PREC_MIN*) to activate the servoing control.

- one exception servoing control of "type 1" indicates that odometric registers are overflowing and implies the *ANIS* Module-task to reconfigure the registers. The servoing control remains active.

- two events (monitored by *OBS_URG*) stop the robot task when

  - execution time is elapsed,

  - the user decides to stop on keyboard.

- head axes limits generated by *LIMITOBS*,

- events on windows image managed by *WINOBS* when :

  visual features are lost,

  visual features will soon disapear,

  a visual obstacle occurs in a window.

In the future, we will be further defined the Robot-task behavior by adding more observers.

These specifications are encoded using the synchronous language ESTEREL ([BER 92]). The compiler transforms an ESTEREL program into a finite determinist automaton. On the resulting automaton automatic proofs can be performed using verification systems (AUTO described in [ROY 90]) and therefore one can be sure of the logical correctness of the program.The Robot-task automaton is illustrated on figure 6.

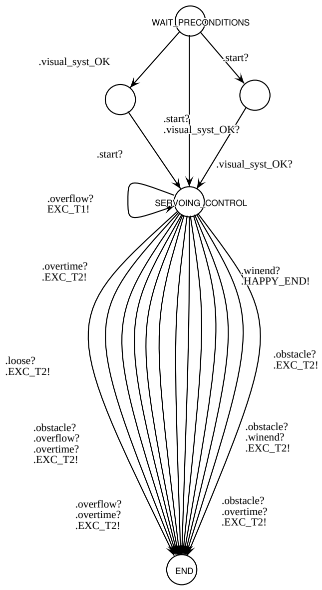

Figure 6. Robot-task automaton

The modelling object of the Module-task which imposes the specified behavior to the Robot-task is an instance of the class "RTA". The Robot-task designer have not to write any ESTEREL program. The automaton code and its interfacing with the environment code is automatically generated in the ORCCAD system using information provided considered events provided by the user. Special considerations have been made at implementation level where safe type communication protocol is used for message passing ....

Robot-task simulation The robotic system, belongs to the physical world, and can generally be described by a set of differential derivate equation in continuous time ; On the other side, the controller works basically in the frame of discrete time. SIMPARC (SImulation for MultiProcesseurs Advanced Robot Controller), see ([AST 92]), is an hybrid simulator from time point of view which mixes both continuous and discrete simulation.

Using the time constrained specification graph of the visual servoing Robot-task specified by the control expert, it is possible to automaticaly generate code for SIMPARC simulation. Simulation results help the user to verify and tune the control law and to choose temporal properties to be assigned to the Module-tasks : activation period, synchronization type.

Robot-task implementation When the Robot-task simulation results are satisfying, an other feature of the designing tool can also generate a downloading code to be executed by the robot system. Module-tasks are VxWorks real time tasks running on the robot CPU board. They communicate between each other using specified synchronization through shared memory.

All the design process has been tested with a cartesian servoing Robot-task and good results have been obtained. Actually we work on system vision integration and on time constrained specification graph of the visual servoing Robot-task. Experimental results are now gathered for tasks like *visual following of a wall*, *visual positionning in front of a door*.

6. Conclusion

In this paper, we both presented the theoretical framework and the implementation tools allowing to validate reactive sensor based approaches applied to mobile robots. We state a robotic action as a set of elementary servoing tasks (continuous time aspect) associated with a logical behavior events driven (reactive aspect). On the first hand, we propose a powerful formalism to build visual servoing tasks allowing to explicitely control the robot in closed loop with regard to the environment. On the other hand, we use synchronous programming approach to solve the reactive aspect of the task. We have also developped an experimental testbed constituted by a mobile robot carrying a two d.o.f head with a CCD camera associated to a dedicated real time image processing system. All the programming aspects are taken into account by the ORCCAD system providing powerfull tools for designing, implementing and checking robotics applications. Presently, the integration of our testbed is almost finished and we expect a complete visual servoing experiment for the end of the year.

References

[AST 92] C. Astraudo, J.J. Borrelly :*Simulation of Multiprocessor Robot Controllers*, Proceedings IEEE Int. Conf. on robotics and automation, Nice, May 92.

[BEN 91] A. Benveniste, G. Berry :*The synchronous approach to Reactive and Real-Time Systems*, Proc. of the IEEE, vol 79, n 9, September 1991.

[BER 92] G. Berry, G. Gonthier :*The synchronous Esterel Programming Language : Design, Semantics, Implementation*, Science of Computer Programming, Vol 19, no 2, pp 87-152, Nov. 1992.

[BRO 86] R. Brooks : *A robust layered control system for a mobile robot*, IEEE Transaction on Robotics and Automation, vol 2, n1, pp 14-33, 1986.

[CHA 90] F. Chaumette : *La relation vision-commande: théorie et applications à des tâches robotiques*, Ph-D Thesis, University of Rennes I, France, July 1990.

[COS 92] E. Coste-Manière, B. Espiau, D. Simon :*Reactive objects in a task level open controller*, Int. Conf. IEEE on Robotics and Automation, Vol 3, pp 2732-2737, Nice, May 1992.

[ESP 87] B. Espiau :*Sensory-based Control: Robustness issues and modelling techniques. Application to proximity sensing*, NATO Workshop on Kinematic and Dynamic Issues in Sensor Based Control, Italy, October 1987.

[ESP 92] Espiau B., Chaumette F., Rives P. : *A New Approach to Visual Servoing in Robotics*, IEEE Transaction on Robotics and Automation, Vol. 8, No. 3, June 1992

[MAR 91] Martinet P.,Rives P.,Fickinger P.,Borrelly J.J : *Parallel Architecture for Visual Servoing Applications*, Workshop on Computer Architecture for Machine Perception, Paris, Dec. 1991

[RIV 89] P. Rives, F. Chaumette, B. Espiau : *Visual Servoing Based on a Task Function Approach*, 1stISER International Symposium on Experimental Robotics, Montreal, 1989

[RIV 92] P. Rives, R. Pissard-Gibollet : *Reactive Mobile Robots Based on a Visual Servoing Approach*, AIS'92, Perth, July 1992

[RIV 93] P. Rives, J.J. Borrelly, J. Gallice, P. Martinet : *A Versatile Parallel Architecture for Vision Based Applications*, Workshop on Computer Architecture for Machine Perception, New Orleans, Dec 15-17, 1993

[ROY 90] V. Roy, R. de Simone :*Auto and autograph*, In R. Kurshan, editor, *proceedings of Workshop on Computer Aided Verification*, New Brunswick, June 1990.

[SAM 91] C. Samson, K. Ait-Abderrahim : *Feedback control of a nonholonomic wheeled cart in cartesian space*, Proceedings IEEE Int. Conf. on robotics and automation, Sacramento, pp 1136-1141, April 1991.

[SAM91b] C. Samson :*Time-varying feedback stabilization of nonholonomic car-like mobile robots*, INRIA research report, n 1515, Septembre 91.

[SAM 90] C. Samson, B. Espiau, M. Le Borgne : *Robot control: the task function approach*, Oxford University Press, 1990.

[SIM 93] D. Simon, B. Espiau, E. Castillo, K. Kapellos :*Computer-aided design of generic robot controller handling reactivity and real-time control issues*, IEEE Trans. on Control Systems and Technology, to appear, fall 1993.

[WEI 84] L. E. Weiss : *Dynamic visual servo control of robots. An adaptive image based approach.* Carnegie Mellon Technical Report, CMU-RI-TR-84-16, 1984.