6th IFAC Symposium on
COST ORIENTED AUTOMATION
(LOW COST AUTOMATION LCA2001)
Berlin, Germany, October 2001

IS VISION THE APPROPRIATE SENSOR FOR COST ORIENTED AUTOMATION?

Friedrich Lange, Gerd Hirzinger

Institute of Robotics and Mechatronics
Deutsches Zentrum für Luft- und Raumfahrt e. V. (DLR)
Oberpfaffenhofen, D-82234 Wessling, Germany
www.robotic.dlr.de/Friedrich.Lange Friedrich.Lange@dlr.de

Abstract: The article points out that a camera is a flexible sensor and that robots can benefit from visual information for different applications. Concerning cost oriented hardware we restrict to standard vision components. We do not need any hardware besides a camera, a low-cost frame grabber, and the robot with its PC-based controller. Software is computationally efficient since only single image rows are evaluated. Our dynamical sensor control architecture distinguishes between robot positional control and refinement of desired positions using vision and / or other sensors, optionally.

Keywords: robot vision, sensors, low cost, high speed, accuracy

1. INTRODUCTION

At first sight the questioner of the title seems to be ignorant since without doubt online evaluation of camera images or sequences is one of the most expensive techniques to explore the environment or to measure the location of objects. Apart from the cost normal visual systems are not able to provide accurate sensor values in time - at least not for feedback control of high speed motion.

Compared to other sensors the image rates of standard cameras are low as they are. In most cases restricted computing power reduces the available control rates to even smaller values. Vincze (2000) sums up the different delay times of usual vision systems and gets a minimum of 60 ms and a maximum of much more than 1 s for one loop. In contrast other sensors can provide data at a rate of more than 1 kHz (see e.g. components in (DLR, 2000)).

Furthermore the attainable accuracy of cameras is limited by low spatial resolution of the images or the evaluated windows of e.g. 64×64 pixels.

On the other side the stiff and precise construction of modern robots as well as their control hardware are expensive. These features are necessary, however, to guarantee high repeatability and absolute accuracy. Accuracy during high speed motion is not even reached. This is an argument for sensors. And there are examples which show that special visual systems are valuable sensors in high performance tasks:

Nakabo et al. (2000) use a high speed camera with an aquisition rate of 1 kHz and a resolution of 128×128 pixels. They succeed in high speed visual servoing. A robot hand is controlled to follow a white ball which is moved randomly. The delay is mainly due to the dynamical behaviour of the robot. The robot reaches the ball and can grasp it when it is held still for a moment.

Hoshino and Furuta (2000) stabilize a triple spherical inverted pendulum by a two-axis robot (3 rigid rods connected with universal joints, the first rod put on the end of a robot arm which moves in the horizontal plane). The upper ends of the three rods are sensed by a camera at a rate of 240 Hz.

Then the whole system is controlled by moving the robot joints.

Gangloff and de Mathelin (2000) use a nonstandard camera, too, with 120 noninterlaced images per second. They design a predictive controller for a robot hand to track a set of printed dots. The reference is varying so that the robot follows a rectangular trajectory with a speed of 5 cm/s when the target is not moved.

Common to all these approaches is that they are able to control a robot by camera images. In contrast to *look - then move* systems, image acquisition and robot motion is performed in real time and simultaneously (*dynamic look and move*). This allows continuous control during task execution. Unfortunately these approaches use nonstandard hardware for image acquisition and / or for image processing. Therefore their application to industrial tasks is too expensive.

If there would be a possibility to use standard (low cost) vision components and to disclaim special hardware for computation, then a camera would be a well suited sensor for cost oriented robotics.

It is accepted that a (high speed and high resolution) camera is more flexible than other sensors on the understanding that there exists a vision algorithm qualified to cope with difficult lighting conditions. With respect to the sensed information, range finders or scanners correspond to only one pixel or one image row respectively. Tactile sensors show the disadvantage of reaction forces to a system which actually should only be monitored, not excited or damped. In addition, visual sensors can measure bigger deviations than force / torque sensors even if those are compliant. We do not state that cameras are the best choice for all applications, but there are tasks in which a camera is superior to other sensors - on the condition of suitable image processing algorithms.

And this is the scope of this paper: We shall show that online vision does not necessarily require expensive hardware. Of course we cannot solve all vision problems with low cost methods. Fortunately, industrial environments are much simpler than real world (outdoor) scenes and, in contrast to exploration tasks, applications in automation deal with a known world in which only some parameters (locations) have to be determined. This allows some simplifications:

E.g., in most cases we have constant lighting conditions. At least aside of windows or skylights there is no difference between night and day. And also if sun has a disturbing effect we can conceal it by artificial lighting. Or we use automatic gain control (AGC) which for standard cameras is available without extra cost.

To measure the location of unknown objects in 3D or 6D, usual methods rely on binocular vision (Malis *et al.*, 2000), in most cases stereo-vision (e.g. (Landzettel *et al.*, 2000)) which requires twice the image processing and additional computing power to solve the correspondence problem. In the case of known objects however, their distance and orientation can be computed by their size and shape in the image. So, in industrial environments mono vision is sufficient (as in (Jörg *et al.*, 2000)). A second camera may be advantageous to increase accuracy as in (Frese *et al.*, 2001) or to be robust against occlusion. Usually it is not required for localization.

For automation a nominal world is always defined and the differences with respect to the actual scene are limited. This often allows to restrict to the evaluation of a small window or, to increase accuracy, the use of telephoto lenses. But one bottleneck remains: If you want to detect features in different directions without rotating the camera, you have to choose your focal length so that all features remain within the visual range. This may result in low angular accuracy. In this case accurate localization is only possible for small distances between camera and features.

In addition to simplifications concerning vision, the robot's hardware and the servo controllers might be constructed more cost effective too, since the absolute positioning accuracy or repeatability is not of such importance if sensors are present. But this goes beyond the scope of this paper.

2. REQUIRED HARDWARE

In this paper we assume an "'eye-in-hand"' setup, where the camera is moved by the robot and the camera is fixed near the endeffector.

The alternative is a setup with locally fixed camera. This is only admissible if the robot actions are limited to a small working space. Otherwise a pan-tilt unit has to be provided, but this is an additional device which needs an extra controller besides the robot controller. Therefore this setup is not more cost-oriented.

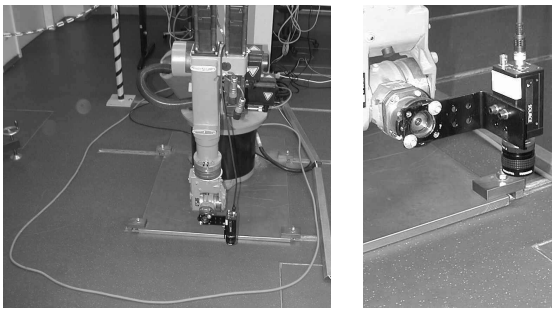

Fig. 1. Experimental setup with endeffector mounted camera

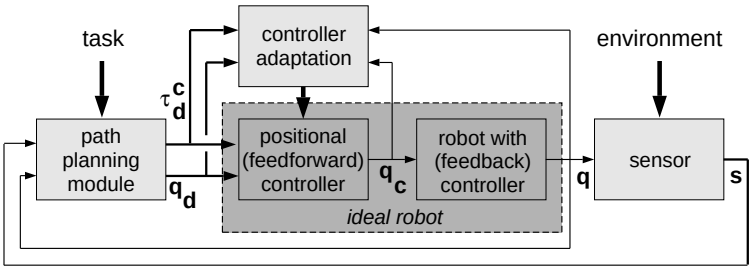

Fig. 2. Splitting up of control to path planning in relation to the sensed environment and to robot positioning in relation to the robot dynamics

Our camera is mounted lateral at the robot flange (see figure 1). This position guarantees minimal occlusion while no big objects are grasped and transported by the endeffector. At the same time the distance to the objects to be seen is minimal. The holding device may be adapted to the size of the endeffector in use. It does not oppose to stiff or compliant force / torque sensors if tactile information is required, too. The weight of the camera including lens is about 350 g which is not worth speaking of for standard industrial robots. So we do not use special miniaturized or microhead cameras with distant electronics which are recommended for space applications.

We use a standard 1/2" monochrome CCD camera with a focal length of 6 mm. At a distance of 300 mm which provides a sufficient depth of focus, the focal length gives a visual range of 240 mm $\times$ 320 mm ($44^o \times 56^o$). With 768 pixels per row this yields a horizontal resolution of 0.4 mm or less in case of subpixel evaluation.

The drawback of standard lenses with small focal length is distortion. Nevertheless we did not supply high precision optics with high refracting glass. Instead we compensate distortion by software (section 4.3).

Our robot KUKA KR6/1 is controlled by the standard configuration, the industrial controller KRC1 which runs on a usual PC platform, in our case with 400 MHz. We do not need any additional processor, neither for image acquisition and evaluation nor for integration of the results to the KRC1 or for displaying the current images. The only hardware component we need besides the standard equipment of the KRC1 and the camera is a frame grabber. As frame grabber we again rely on low cost products with small FIFO only, as we do not need any hardware supported computations. The only use of the grabber is to load standard PAL fields of 288 $\times$ 768 pixels in the PAL field rate of 50 Hz to the RAM.

3. SOFTWARE ARCHITECTURE

KUKA has used VxWorks for real-time control and Windows95 as low priority man-machine interface. We adopted this approach and implemented our controller under VxWorks while the frame grabber is accessed by Windows.

In the KRC1 a sensor interface is available which provides the actual joint positions $\mathbf{q}(k)$ (or cartesian values) at the beginning of each interpolation step k and accepts new setpoints (commands $\mathbf{q}_c(k)$) at the end of the step (figure 2). In our case these setpoints are composed by the stored path, terms to compensate for deviations due to the robot dynamics, and some path modifications induced by vision.

The vision algorithms run under Windows. Only the result of about 50 Bytes is sent to the VxWorks part of the system memory. Data transfer between the two operating systems is socket like. Because of the cycle time of 20 ms for the acquisition of the image fields and the cycle time of 12 ms of the path interpolator, communication takes place asynchronuously (see section 4.3).

Parallel to image evaluation the camera output is displayed by a different process. This is for debugging by the operator only and can therefore run in a small window. This display combines the monochrome camera image with coloured markers representing the detected feature locations (figure 3). For further analyses limited time sequences can be recorded and replayed in slow motion or single step mode, with full resolution if required.

4. DYNAMICAL SENSOR CONTROL ARCHITECTURE

For control we use the architecture of (Lange and Hirzinger, 2001) defining an ideal robot which is able to follow a desired path $\mathbf{q}_d$ without deviations and without time delay. The control task is then splitted into the realization of this ideal system and into the sensor based generation of the desired path (figure 2). The realization of the ideal robot is independent of the task or the sensor equipment in use. On the other side the path planning algorithm needs no information about the dynamics of the robot. We think that the independence of the two blocks justifies the use of a *position based* approach instead of an *image based* method.

4.1 *Interface of the ideal robot*

Input to and output from the ideal robot are positions, either in joint coordinates or in cartesian space. The demand to follow without time delay calls for an extended input structure that includes something besides the desired current setpoint $\mathbf{q}_d(k)$. Possible extensions are the desired velocity and the desired acceleration or, what we prefer, the desired setpoints at subsequent timesteps $\mathbf{q}_d(k+i)$. If contact forces are present that affect the robot, we also include desired current and future contact forces $\tau_d^c(k+i)$, if available. For a well suited controller these variables are sufficient to control the robot ideally if the robot parameters are identified and if the desired path is known for an adequate number of timesteps ahead. For the number of timesteps n_d we recommend to cover the main time constant of the robot including delay times of the sensor interface. This sums up to 100-200 ms.

This interface has been successfully used in other applications too, e.g. for fast execution of nonlinear and noncircular paths (Lange and Hirzinger, 1996*b*) or for force tracking along an unknown contour (Lange and Hirzinger, 1996*a*).

4.2 *Implementation of the ideal robot*

The use of a special controller to realize an ideal robot is optional within our sensor control architecture. We also executed the experiments of section 6 using the industrial positional controller of KUKA only, and the result was still acceptable.

If a special controller shall be used to implement an ideal robot, there are different setups possible. Suitable are all controllers which compensate unwanted dynamical effects. To do so the temporal derivativies of the desired position or the sequence of future desired sampled positions have to be processed. The first is done by a computed torque approach (An *et al.*, 1988), the latter is the approach used for the experiments of this paper (Lange and Hirzinger, 1996*b*) which can be extended for elastic robots (Lange and Hirzinger, 1999).

For historical reasons our control architecture uses the robot control system as it is and adds a feedforward controller which affects the system via the provided sensor interface. The feedforward controller is adapted to the robot dynamics. Since the ideal robot is independent of the task to be executed, it is sufficient to perform the adpatation once, at the installation of the robot. Additional adaptations can compensate for abrasion.

The use of the underlying high rate robot control loops allows the feedforward controller to act at low sampling rates. In most applications a reduced controller setup is sufficient which every 12 ms linearly combines subsequent desired positions to compute the next command $\mathbf{q}_c$. This is a kind of predictive control in which only desired values are processed, i.e. there is no feedback of the measured values. This is approved since statically the internal controllers compensate all positional disturbances.

4.3 *Sensor based generation of the desired path*

The advantage of the proposed architecture compared with direct feedback of sensor data is that no sensor-specific controller has to be designed. The path planning algorithm simply has to determine the desired positions.

The input to the ideal robot in form of a part of a desired trajectory, not of a single position, makes the system insensitive to delays in the communication. Even asynchronuous sensing is allowed. For the case of a camera every new field is processed and used in the next control cycle to refine or to extend the part of the trajectory. So for a sufficiently big visual range the required section of the next n_d timesteps is always defined, also if sensing or communication is interrupted for some milliseconds. This allows the vision system to run at low priority thus guaranteeing the performance of the positional controller.

If the visual range is exceeded we can extrapolate the desired path just by keeping the differences between detected and nominal locations constant. This corresponds to a displaced workpiece the shape of which is correct.

If a nominal path is not available, e.g. if an unknown contour has to be tracked as in (Lange and Hirzinger, 1996*a*), the desired path has to be predicted using e.g a first order extrapolation of the so far sensed contour.

For fixed camera systems the desired position of the endeffector is computed by the difference between the image locations of endeffector and target (*end point closed loop*).

For eye-in-hand systems, the target location can be determined relative to the camera and the camera location is given by the robot kinematics and the joint angles at the time-instant of the exposure. Because we do not use cameras with trigger input, this time-instant can only be fixed to about 10 ms. So at the first glance for motion with 1 m/s we expect an achievable accuracy for the target location of not more than 10 mm. Reconsidered, the temporal uncertainty has no effect on the feature recognition since the image is consistent, leaving out changes during the exposure time. Further, for high speed, accurate localization is of interest only

perpendicular to motion. And these coordinates are orthogonal to the uncertainty caused by a roughly known time-instant of the exposure. E.g. when following a line, only progress is uncertain, not the displacement from the line.

Since at least for future tool positions only the target can be seen (*end point open loop*), the mapping from the image to the real world has to be calibrated. Especially knowledge of the orientation is required for high speed motion. Then features at the image border will be reached by the tool center point (TCP) some timesteps later. So the absolute location of the features is required precisely to allow refinements of motion by feedforward. We use a tool-box (DLR, 1999) that combines identification of lens distortion and determination of the camera location relative to the TCP.

4.4 *Stability*

The dynamical sensor control architecture preserves stability as long as the positional robot feedback controller is stable. Sensor and position feedback in figure 2 seem to affect stability, but actually the signals describe the location of the target, not of the robot. Stability is touched only when the robot position itself is fed back. This may be the case if the transformation from sensor values (feature locations in the image plane) to desired robot positions is totally wrong (details are in (Lange and Hirzinger, 1996*a*)) or if no features can be localized.

Besides, intentionally or unintentionally (due to noise) high bandwidth desired paths may excite the elastic modes of the robot. Then the assumed camera position is incorrect. In this way the robot motion may be fed back to the input, too. Therefore oscillations have to be avoided.

5. COST-EFFECTIVE VISION

Computationally inexpensive evaluation of images requires that not all pixels are used. The restriction to e.g. 28 of 288 image rows or 76 of 768 image columns already reduces the costs by a factor of 10. Nevertheless the resolution is unchanged along these horizontal or vertical image lines respectively. If e.g. a horizontal image coordinate of a feature has to be detected and the approximate location of this feature is known then it is sufficient to scan a window of, say, 200 pixels within 1 image row. The computational cost for this is minimal.

If the image has to be rectified because of lens distortion, rectification can be restricted to the pixels which are used thereafter, in our example to 200 pixels, instead of a whole field of 200,000.

Fig. 3. Displayed camera image with (coloured) markers representing the used window and five detected line points

The localization of features is proposed as one or more line searches within given image rows, i.e. in horizontal direction. In this case the resolution corresponds to one pixel. In the other case of inclinated or vertical searching, the accuracy is worse because in field-mode either even or odd image rows are available. In addition subpixel evaluation is simpler for a search direction that is aligned to the image coordinates.

In section 4.3 we argued that accuracy in the direction of motion is less important. Therefore we propose to align the camera such that motion is executed with respect to the vertical image coordinate. But this is not obligatory.

Because of the horizontal search, localization with respect to this direction is the simpler case. But features as circles or circular disks can be used to measure the camera location in more degrees of freedom, in 5D theoretically. Resolution is identical with respect to the horizontal and the vertical image coordinate since the position of the image row in use is always defined accurately. There is no vertical uncertainty of 2 pixels. Solely every second row is missing. So each detected line point is located with full resolution. Therefore all features that differ from a horizontal line are useful for localization.

Regarding subpixel evaluation we recommend a camera with automatic gain control (AGC). In addition, in our experiments we adapted the threshold to the local brightness of the image row in the used window.

6. EXPERIMENTS

As experiments we first considered line following (as in (Lange *et al.*, 1999)) which is similar to industrial applications as laser cutting or applying

glue where the target motion is defined relative to one or more edges and the positions or the shapes of the edges differ from execution to execution. Such task should be executed with high speed insuring high accuracy at the same time.

A similar experiment is executed with the cost oriented setup of this paper (see figures 1 and 3). In our experiment a single line is tracked, represented by a polynom which is computed from 5 points. After loading of the necessary functions, the computing power of the single processor which at the same time is used for robot joint control, easily achieves the required performance of evaluating 50 fields per second so that it is additionally possible to display the scene online. The mean control error is 0.6 mm for a top speed of 0.7 m/s.

Other applications are pick-and-place operations from coarsely known pickup positions to coarsely positioned insertion places. In this case the advantage of real-time vision is that during approaching no extra stop is required for sensing and that calibration errors can be compensated by refining the target position from step to step. At the end, speed is zero so that misalignment of the camera has no more effect.

7. CONCLUSION

We presented a standard vision system running on a standard robot control computer without additional hardware support. Such components are available for less than 1000 € or 1000 US $. Nevertheless the system was able to localize target positions in video rate. Using a dynamical sensor control architecture the robot could follow a seen line with less than 1 mm positional error in spite of a speed of 0.7 m/s. Other potential applications are on-the-fly pick-and-place tasks with coarsely known positions. The profit of such rapid and flexible sensing to factory automation exceeds the cost by far.

8. REFERENCES

An, C. H., C. G. Atkeson and J. M. Hollerbach (1988). *Model-Based Control of a Robot Manipulator*. The MIT Press. Cambridge, Massachusetts, London, England.

DLR (1999). http://www.robotics.dlr.de/VISION/Projects/ Calibration/CalLab.html.

DLR (2000). http://www.robotics.dlr.de/MECHATRONICS.

Frese, U., B. Bäuml, S. Haidacher, G. Schreiber, I. Schäfer, M. Hähnle and G. Hirzinger (2001). Off-the-shelf vision for a robotic ball catcher. In: *Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems*. Maui, Hawaii.

Gangloff, J. A. and M. F. de Mathelin (2000). High speed visual servoing of a 6 DOF manipulator using MIMO predictive control. In: *Proc. IEEE Int. Conference on Robotics and Automation*. San Francisco, California. pp. 3751–3756.

Hoshino, T., H. Kawai and K. Furuta (2000). Stabilization of the triple spherical inverted pendulum - a simultaneous design approach. *at (Automatisierungstechnik)* **48**(12), 577–587.

Jörg, S., J. Langwald, J. Stelter, G. Hirzinger and C. Natale (2000). Flexible robot-assembly using a multi-sensory approach. In: *Proc. IEEE Int. Conference on Robotics and Automation*. San Francisco, California. pp. 3687–3694.

Landzettel, K., B. Brunner, G. Hirzinger, R. Lampariello, G. Schreiber and B.-M. Steinmetz (2000). A unified control and programming methodology for space robotics applications. In: *31st International Symposium on Robotics*. Montreal, Canada.

Lange, F. and G. Hirzinger (1996*a*). Learning force control with position controlled robots. In: *Proc. IEEE Int. Conference on Robotics and Automation*. Minneapolis, Minnesota. pp. 2282–2288.

Lange, F. and G. Hirzinger (1996*b*). Learning of a controller for non-recurring fast movements. *Advanced Robotics* **10**(2), 229–244.

Lange, F. and G. Hirzinger (1999). Learning accurate path control of industrial robots with joint elasticity. In: *Proc. IEEE Int. Conference on Robotics and Automation*. Detroit, Michigan. pp. 2084–2089.

Lange, F. and G. Hirzinger (2001). A universal sensor control architecture considering robot dynamics. In: *Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*. Baden-Baden, Germany.

Lange, F., J. Langwald and G. Hirzinger (1999). Predictive feedforward control for high speed tracking tasks. In: *Proc. European Control Conference*. Karlsruhe, Germany.

Malis, E., F. Chaumette and S. Boudet (2000). Multi-cameras visual servoing. In: *Proc. IEEE Int. Conference on Robotics and Automation*. San Francisco, California. pp. 3183–3188.

Nakabo, Y., M. Ishikawa, H. Toyoda and S. Mizuno (2000). 1ms column parallel vision system and it's application of high speed target tracking. In: *Proc. IEEE Int. Conference on Robotics and Automation*. San Francisco, California. pp. 650–655.

Vincze, M. (2000). Dynamics and system performance of visual servoing. In: *Proc. IEEE Int. Conference on Robotics and Automation*. San Francisco, California. pp. 644–649.