Flexible Automation of Cell Culture and Tissue Engineering Tasks

Alois Knoll,*,[‡] Torsten Scherer,[‡] Iris Poggendorf,[†] Dirk Lütkemeyer,[†] and Jürgen Lehmann[†]

Department of Cell Culture Technology, Faculty of Technology, University of Bielefeld, POB 100131,
33501 Bielefeld, Germany, and Department of Informatics, Technische Universität München,
Boltzmannstrasse 3, 85748 Garching, Germany

Until now, the predominant use cases of industrial robots have been routine handling tasks in the automotive industry. In biotechnology and tissue engineering, in contrast, only very few tasks have been automated with robots. New developments in robot platform and robot sensor technology, however, make it possible to automate plants that largely depend on human interaction with the production process, e.g., for material and cell culture fluid handling, transportation, operation of equipment, and maintenance. In this paper we present a robot system that lends itself to automating routine tasks in biotechnology but also has the potential to automate other production facilities that are similar in process structure. After motivating the design goals, we describe the system and its operation, illustrate sample runs, and give an assessment of the advantages. We conclude this paper by giving an outlook on possible further developments.

Introduction

With the exception of a few flow-process oriented high-volume production sites, biotechnology plants critically depend on human interaction with the process, e.g., for material and cell culture fluid handling, transportation, operation of equipment, and maintenance. For the most part, the tasks carried out by humans in this context are highly repetitive and hence both error-prone and unattractive. To obviate the need for human intervention with the aim of maximizing efficiency and minimizing the risk of failure, error, and contamination in biotechnology environments (but also in a large variety of structurally similar settings, e.g., for tissue engineering), a very promising way is to introduce robots (fully programmable manipulators) into the plant process and transfer all suitable tasks to them.

Until now, the predominant uses of robots have been routine handling tasks in the automotive industry. In those settings they have become an essential and very successful part of the production process. There are, however, a number of deficiencies inherent to these robots that have prevented their use in the unstructured scenarios found outside of the typical automotive production plant. These deficiencies are deeply rooted in the hardware and software designs of these robots and pertain to their flexibility, ease of operation, and safety.

The general requirements to be met by a robot system to be applied successfully to biotech tasks are manifold; in our view the three most important ones are characterized as follows:

Flexibility. The robot system should be capable of adapting to changing environments, i.e., changes in the layout of the plant, the equipment used, the actual tasks to be performed, etc. This necessitates the use of *complex sensors*, i.e., the operation of the robot must be highly sensor-based. Moreover, if at all necessary, the process itself should be modified at most marginally: all of the standard equipment should stay in its place, and the robot should be able to move from one “work station” to another. We call such types of robots *mobile manipulators*.

Programmability. Standard industrial robots require experts not only to program them to perfom a specific task but also for all modifications of these programs. For more widespread use of service robots, however, the programming efforts must be reduced to a minimum. On one hand, this reduction can be achieved by observing the enviroment with powerful sensors capable of automatically adapting manipulation operations to the situational context. On the other hand, it is also very important to make the (textual) programming of the program structure as abstract and readable as possible, so that even novice users can get the robot running quickly with a steep learning curve.

Safety of Operation and Coexistence with Humans. The robot system should coexist with humans in the same plant: both should not only move about in one room freely but should also cooperate in the sense that the human can direct the robot (not) to do certain things, that the human can support the robot to perform operations (and vice versa), etc. This also implies that all constituent parts of the process must not be modified. Instead, it must be possible that they be operated both by the robot and the human operator’s hand. It goes without saying that the robot must not under any circumstances hurt humans.

In the sequel we describe a robot-based solution for flexible automation, which was developed with these general requirements in mind. To justify our claim of this research route having a large potential, we present a

* To whom correspondence should be addressed. Tel: ++49 (0)-89 289-18106. FAX: ++49 (0)89 289-18107. E-mail: knoll@in.tum.de.

[†] University of Bielefeld.

[‡] Technische Universität München.

10.1021/bp049759v CCC: $27.50
Published on Web 11/04/2004

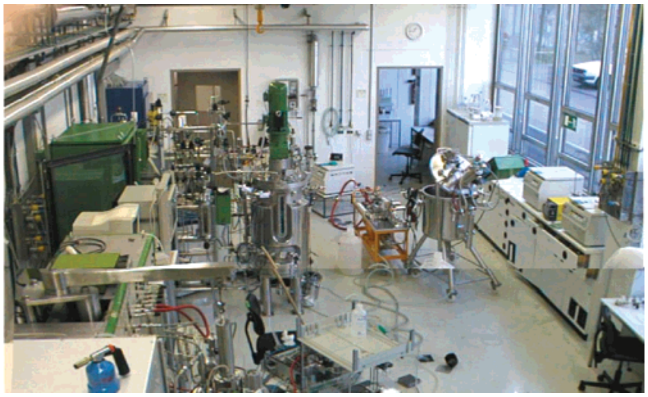


Figure 1. View of the biotechnology pilot plant that was used for carrying out the test fermentations with automated sample management.

complete application: *sample management* in biotechnology (the analogies with other applications, in particular tissue engineering, are readily established). With our robot solution, this process has not only been automated robustly, it has also been tested thoroughly over the past years and demonstrated on various occasions.

It is important to note that this application is by no means the end of a very promising development, rather a first step. The approach offers the potential to completely automate a large spectrum of handling, controlling, and maintenance tasks in many areas.

Problem Statement

The process of sample management, as automated here, is an essential part of the biotechnological process of mammalian cell cultivation for the production of biopharmaceuticals. It is crucially important for monitoring the culture and for determining the optimal harvest time. The main steps of this process are as follows:

- Take a sample from the bioreactor by filling a small amount of cell culture fluid into a vial;
- Transport the vial to the cell counting device, transfer a fraction of the cell culture fluid into it, start the device, count the cells, and determine their viability;
- Separate the cells from the broth using a centrifuge (after having moved the sample to the centrifuge);
- Store the aliquot of the cell-free supernatant in a freezer for further measurements.

These actions include the handling of multiple types of tubes, pipetting of liquid from/into the tubes, feeding them to various devices, and operating the devices. Figure 1 shows a complete view of the laboratory at the University of Bielefeld, which was used throughout the development of the system (i.e., the “habitat” of the robot).

A normal batch-type cultivation of mammalian cells usually takes up to 2 weeks, whereas continuous cultures might run for several months. During this time human personnel must be present for sample management, process supervision, and to perform necessary process changes. Because this includes both nights and weekends, it presents a high cost factor. Human personnel also introduce unpredictable errors when judging the sample, depending on their training and fatigue.

Attempts to automate sample management with online sample analysis have so far focused on designing special and complex machines that are directly attached to a bioreactor. These machines are both expensive and inflexible. Even the slightest change in the analytical process may make them useless or require expensive modifications. They also increase the amount of equipment that has to be kept sterile in order to avoid contamination during bioreactor operation.

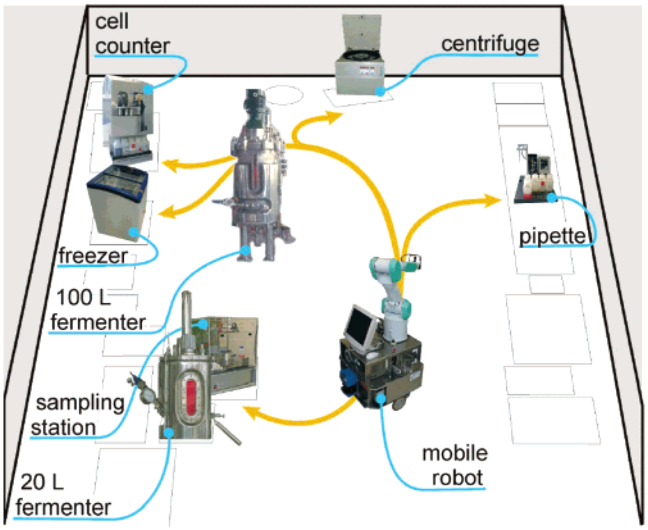


Figure 2. Symbolic (“Process”) view of the laboratory of Figure 1. It only shows the devices to be controlled and operated by the robot, as well as the trajectories of the robot at work.

There is, however, a large range of semiautomatic stand-alone analytical devices for most culture parameters available. For example, the CEDEX cell counter (*11*) used in our setup automates the cell count and classification. It uses a computer vision system to evaluate a sample dyed with the standard trypan blue method, yielding a fast and reliable analysis (*13*). Being semiautomatic, it normally has to be loaded/unloaded and operated by human personnel.

The approach described in the rest of this paper is based on a mobile manipulator, i.e., a programmable highly dextrous robot arm mounted on a wheeled mobile platform with high maneuverability. In its current form it is perfectly suited to standard laboratory rooms and the equipment in use there as outlined below, but as all parameters are scalable, i.e., platform geometry and size, gripper forces, etc., it can be adapted to a large range of needs.

This robot system was not only designed to automate the *entire* sampling process, it also makes it more consistent (*17*). As mentioned above, it is composed of a mobile platform that navigates freely in the laboratory and a robot arm to carry the sample and operate devices (see Figure 2). It presents a completely new approach in that it uses *standard laboratory equipment* with only a minimal amount of modifications, if any. The key characteristics of this system are as follows:

- **Ease-of-use.** The system can be programmed in a very comfortable way to new tasks (see below) and is capable of automatically adapting to changes in the environment (e.g., objects and devices that have been moved away from their original position). The system software architecture has been so designed as to allow for an easy alteration of system parameters by nonexpert personnel.
- **Continuous Operation.** The system may be used without any interrupts over unlimited periods of time. It is battery powered and docks to an automatic recharger whenever there is time between performing the individual tasks.
- **Robustness and Fault-Tolerance.** As a result of its adaptive properties based on various sensors, the system can easily handle variations in the environment.

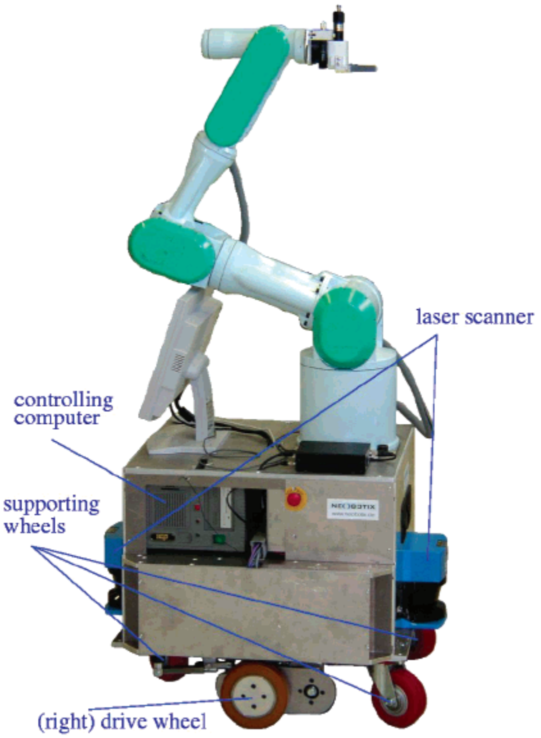


Figure 3. Mobile platform and robot arm.

It also detects humans walking in the plant, stops while they are in the way, and resumes work afterward. All manipulations are supervised; if any step fails in the process, the robot stops immediately.

- **Operation in Hazardous Environments.** The robot can operate in sterile and biohazardous environments. It can also be used in areas that are inaccessible to humans and can take over potentially dangerous missions.

The potential of the whole system for full validation as an intrinsic part of the process is also of high importance. The same goes for the obvious possibility to *record all operations* on all objects for purposes of documentation.

System Details

The whole system was so designed as to be able to use existing devices for automating processes and to allow human personnel to utilize them for other tasks while the robot is idle, thus eliminating the need for duplicate equipment. The system also introduces a sterility barrier by not being fixedly connected to a reactor and therefore minimizes the risk of contaminating the reactor. Instead, a steam-sterilizable sampling system directly connected to the bioreactor is used to fill a sample into a tube, which is then carried to the different analytical devices by the robot. The system consists of a Mitsubishi PA-10 robot arm (*4*) mounted on a mobile platform (*5*) as shown in Figure 3. The platform is equipped with a differential drive with odometers, a gyro compass, two SICK LMS-200 laser range finders, and a standard PC operated under Linux. The laser range finders measure the distance to obstacles in a range of 180° each and detect special retro-reflecting marks used for localization. The arm itself is equipped with a wrist-mounted force/torque sensor (FTS), a microhead color camera, and an electric parallel yaw gripper as in Figure 4. The arm is connected to the PC via an ARCNET network and controlled at joint controller level by a modified version of RCCL (*1*).

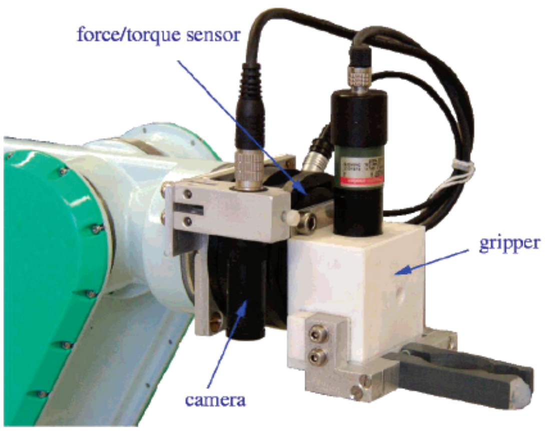


Figure 4. Robot arm tool.

The kinematic redundancy of the system is solved by strictly separating between mobile platform and arm—each is a separate subsystem with its own sensors and positioning strategies described in the next sections.

Control of the Platform

The basis for the successful manipulation of the devices with the robot system is a precise positioning of the mobile platform, for which three prerequisites have to be met: (A) The exact global position and orientation of the platform have to be determined (the localization problem). (B) A path from the current robot position to a goal position has to be found (the navigation problem). (C) A control mechanism has to be implemented that moves the robot according to the computed path (the motion execution problem).

A. Localization. Experiments by Gutmann et al. (*7*, *8*) showed that Kalman filtering techniques yield the precise results for solving the localization problem. In our work an extended Kalman filter (EKF) by Schmidt (*9*) is used. The system-state vector $\vec{x}_t$ contains the platform's angular wheel velocities, its Cartesian position and velocity, and the positions of the i laser reflector marks. These reflector marks serve as landmarks with a known global position in the robot's workspace. The complete state vector $\vec{x}_t$ is

$$\vec{x}_t = (\omega_{\mathrm{R}}, \omega_{\mathrm{L}}, \dot{x}, \dot{y}, \dot{\phi}, x, y, \dot{\phi}, x_{f1}, y_{f1}, ..., x_{fi}, y_{fi})_t^T$$

Apart from the state vector the EKF also uses a measurement vector $\vec{z}_t$ containing all available sensor information. This includes the distances and angles to the reflectors received from the laser range finders, the angular velocities of the drive wheels as reported by the odometers, and the rotational velocity of the platform as measured by the gyro compass. The measurement vector $\vec{z}_t$ is

$$\vec{z}_t = (\omega_{\mathrm{R}}, \omega_{\mathrm{L}}, 3\varphi, d_{\mathrm{f1}}, \alpha_{\mathrm{f1}}, ..., d_{fi}, \alpha_{fi})_t^T$$

The EKF thus merges sensor information of very different accuracy and can give an estimate of past, present, and future system states. In our system it provides a position estimate up to 37 times, but at least about 20 times per second.

B. Navigation. The navigation problem is addressed by the approach discussed by Latombe (*10*). It uses the A*-algorithm (see ref 12) to search the shortest path from the current position to a desired goal position in a tangent graph. The tangent graph is computed from a map using simple polygons to represent static obstacles. Dynamic

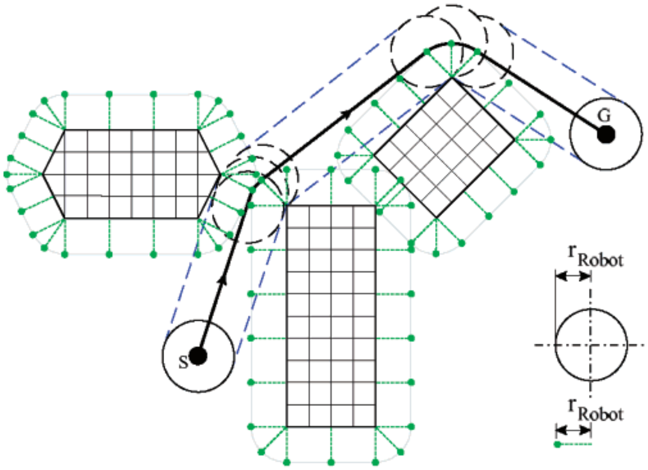


Figure 5. Expanded map with path. The three obstacles are expanded by the radius of a circle that surrounds the robot.

obstacles are not covered by this approach; they are dealt with by collision avoidance.

To use this approach, the representation of the robot in the map has to be shrunk to a point, while all obstacles have to be expanded by the same radius. Using this expanded map, a tangent graph is built, which is the set of straight lines that connect all polygon vertices without going through any polygon body.

The *A**-algorithm returns the combination of those tangents that form the shortest path from start to goal. An example can be seen in Figure 5, showing a path from a point S to a point G.

C. Motion Execution. Motion execution does not generate any behaviors (such as “avoid obstacles”) but strictly follows the computed path. If the platform encounters an obstacle it stops and waits for it to move or to be moved away (after notifying a human operator, if necessary). This may seem slow and inflexible, but it makes the platform predictable and verifiable, features that are of great importance in an environment with human presence in which a machine has to meet several safety standards.

Each segment of the path is taken as a desired trajectory. A PI-controller is used to stay on the trajectory while going toward the target with a one-dimensional trapezoidal velocity profile (*2*). The motion is brought to rest at a line perpendicular to the trajectory through the target point. This allows overshooting of the mobile platform along the trajectory but has the advantage of keeping the platform from getting trapped in a potentially endless loop of trying to reach a small catch radius around the target with a desired orientation. Instead it just stops and reports the deviation to the arm controller for compensation. This compensation is done with the help of visual fine-positioning, which is discussed in the next section.

Visual Fine-Positioning

Although the robot arm may compensate for known errors in the positioning of the mobile platform, other errors remain. These include inaccuracy of the mobile platform’s localization due to noise in the sensor measurements, as well as the unpredictable influence of human personnel having used and possibly moved the equipment. However, a very high positioning accuracy of the arm tool of only about 1 mm is an indispensable prerequisite for successful robot manipulations.

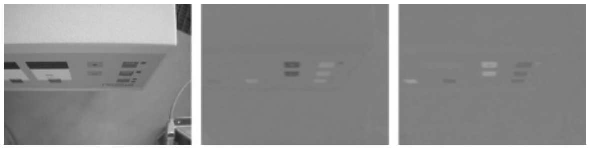


Figure 6. Y, U, and V channel (from left to right) of a CCIR-601 YUV color image (original images, not contrast-maximized).

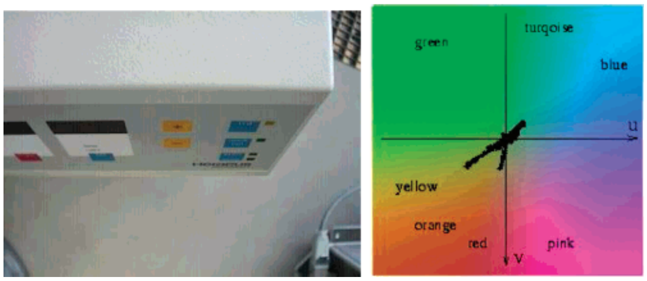


Figure 7. Sample image of the centrifuge (left) and its colors used in the U/V plane (right). It can be seen that only a fraction of the full range of color saturation is used.

For those errors that cannot be dealt with by the compliance control implemented through the force/torque sensor, a color vision system is used for compensation in a look-and-move strategy.

A. Color Vision. TV cameras provide natural and very comprehensive information. However, they require a very large amount of processing to extract the essential information. It is therefore desirable, if not necessary, to remove unwanted information in advance. Since our objects either have colored regions or can be easily tagged with a colored label, we chose to employ a color-based approach to detect objects by searching for known colors. It is hence of high importance to select the correct color representation for our given environment and task. Frequently, the RGB color space is used for reasons of simplicity, but it has the disadvantage of mixing color and brightness information. In contrast, the standardized CCIR-601 YUV color representation (*15*) separates the brightness (Y) from the color (U/V) information and is therefore much better suited to our purposes. Another advantage of YUV is that it is the native “S-VHS” video signal format and can therefore be processed by the frame grabber, with no need for additional conversion. Its only disadvantage is that the color information in a YUV image is encoded with less bandwidth than brightness information and therefore has a lower signal-to-noise ratio. Figure 6 shows the Y, U, and V channels of a sample image.

One approach to detect colored regions is to search for their edges in the U/V images. Classic edge detection methods based on differentiation of the gray values of the image matrix (like the elementary Sobel filters) show poor behavior on signals with a low signal-noise-ratio. Other approaches such as the SUSAN detector by Smith in ref 6 usually perform better in these cases. Because we do not need the region’s shape or edges for the classification we instead chose a *region growing* approach.

The U/V images can be transformed into a diagram showing those colors in the U/V plane that are found in the image. Figure 7 is such a diagram. In this diagram the color *value* is represented by the angle of a vector from the center point into the plane, and the color *saturation* by its length. This value/saturation representation is actually similar to the HSV color format, which may also be used as input format.

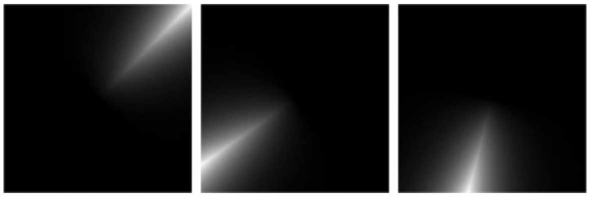


Figure 8. Similarity measure of the colors “blue”, “yellow”, and “red” of the U/V space in Figure 7.

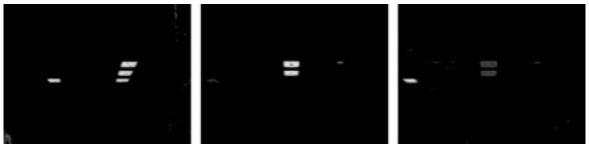


Figure 9. Similarity of pixels to the colors “blue”, “yellow”, and “red” (from left to right) of the image in Figure 7. The colors “yellow” and “red” are similar enough to still yield a low ranking for their counterparts.

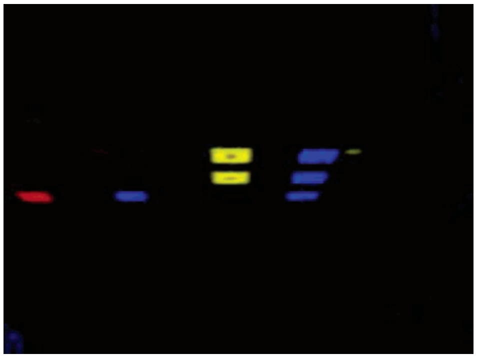


Figure 10. Combined similarity image.

Searching for known colors now means looking for pixels along a vector of a known angle α.

First, the U/V plane is rotated by α:

$$\vec{r} = T_{\mathrm{ROT}(z,\alpha)}\cdot\vec{p}_{\mathrm{uv}}$$

In this rotated coordinate system a similarity measure

$$s = r_x/r_x^{\max}\cdot\exp(-c\cdot|r_y|/|r_y|^{\max})$$

is defined to yield a high ranking for pixels with a high positive r_x-value and a near-zero r_y-value. This measure is more tolerant to accepting variance of the color saturation than variance of the color value, as can be seen in Figure 8.

The measure is then applied to the known colors of objects, yielding similarity “images” of colors as in Figure 9. An example of how the situation with most brightness information removed looks like for the robot is given in Figure 10.

These similarity images are used to find the best-matching (brightest) pixel. Around this pixel a region is built using a *seed fill* algorithm down to a threshold of similarity. The regions determined in this way are stored in a list.

B. Model Matching. These regions are then reduced to their color and *center of gravity* (COG) to build a model of the image scene, which is matched against stored models of objects.

As opposed to learning-based approaches such as neural networks or fuzzy controllers (*14*) this model-based approach needs only one training image. This is particularly important if it is not possible to obtain images of an object from all perspectives, e.g., because a lid is obstructing a part of the workspace.

The matching of the database models against the actual object images is done by shifting and translating the model and then looking at an assignment of COG pixels. Here, allowing only an *xy*-translation and *z*-rotation of a two-dimensional model, the relation between each image pixel $\vec{p}_{\mathrm{i}}$ and its model pixel $\vec{p}_{\mathrm{m}}$ can be described as

$$\begin{bmatrix} \cos\alpha & \sin\alpha & t_x \\ -\sin\alpha & \cos\alpha & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x_m \\ y_m \\ 1 \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix},$$

which can be rearranged to

$$\underbrace{\begin{bmatrix} x_m & y_m & 1 & 0 \\ y_m & -x_m & 0 & 1 \end{bmatrix}}_{M_m} \cdot \underbrace{\begin{pmatrix} \cos\alpha \\ \sin\alpha \\ t_x \\ t_y \end{pmatrix}}_{\vec{u}} = \underbrace{\begin{pmatrix} x_i \\ y_i \end{pmatrix}}_{\vec{p}_i}.$$

Combining the equations of at least two pixels of a complete model yields an overdetermined equation system (we take sin α and cos α as linearly independent for simplification), which can be solved with the pseudo inverse

$$M^{-1} \approx (M^T M)^{-1} M^T$$

to yield the optimal vector $\vec{u}$ of unknowns in the sense of least-squares error (LSE). The LSE

$$e = |M_{\mathrm{m}}\cdot\vec{u} - \vec{p}_{\mathrm{i}}|$$

can in turn be used to find the correct model and pixel assignments.

As can be seen in Figure 11, this approach does not deal with perspective effects caused by displacements and/or lens errors. A 3D model to compensate for these effects has been tested but has been found to have, contrary to the 2D model, local minima. Therefore, only a 2D model is used. Despite the systematic error thus introduced, the approach still allows safe classification of our devices (see Figure 12).

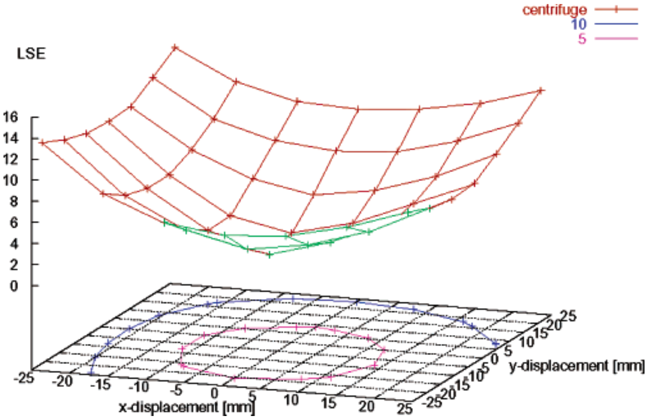


Figure 11. LSE changes caused by perspective effects at different displacements. The circles labeled “5” and “10” show isobars of the error function.

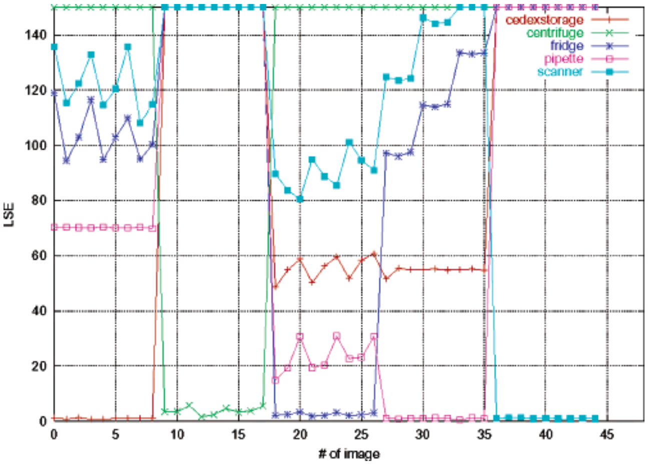


Figure 12. LSE allows an unambiguous classification of different devices with different displacements. Error values have been clipped at 150.

C. Illumination Invariance. The vision system uses no brightness information and is therefore tolerant to most changes in the illumination that affect the brightness. Even changes in the illumination chromacity caused by bright sunlight shining into the otherwise artificially illuminated laboratory are tolerated. Figure 13 shows that the amount of noise in the color data increases and the quality of the resulting regions decreases with worsening illumination conditions. The matching, however, stays largely unaffected by this, because it is only done on the COGs of the regions. Only if the scene should become so dark that complete regions are lost will the classification fail.

Robot Control

The robot acts as a server in a (wireless) TCP/IP network, waiting for the process control system to request its services. It offers a set of high-level functions such as "*fetch a tube*", "*hold a tube under the pipet*", or "*place a tube in the centrifuge and start it*".

A. Sequence Scripts. The sequence of commands and parameters, mostly homogeneous transformations describing spatial relationships, required for each high-level function is stored in a central database and is reread each time the function is invoked. Because they are stored as ASCII text, they can be easily changed.

It has proven to be impossible to use an existing script language/interpreter (such as tcl/tk, perl, python, etc.) and still have all the functionality in the script. This is because of the need to access hardware and/or to have realtime capabilities, e.g., for force control. Instead, we chose to have a set of complex built-in functions that can use the full power of C++ and a realtime operating system and only a rather trivial and custom script language. This lack of complexity in the script language in turn allows operators with comparably little training to do changes.

B. Script Commands. Script commands offer textual access to routines implemented in C++ in the main program. They represent a simplified approach to the full functionality of RCCL (*1*) for arm control, the vision system, and the mobile platform. Only those aspects needed to allow easy adaptation are used in each case. An excerpt of the set of script commands is shown in Table 1.

C. High-Level Functions. With these script commands the set of high-level functions is realized. These functions can operate on a most diverse range of "robot-aides", in our case:

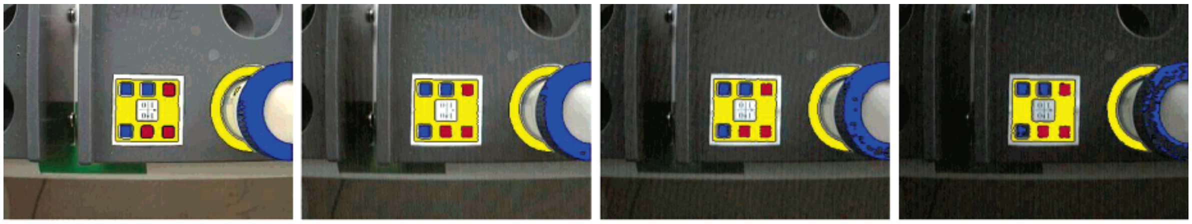
- a sampling device, in which a tube has to be placed and secured while it is being filled with the sample,
- a pipetting device, basically a needle, under which different types of tubes have to be held at different depths,
- a centrifuge, where the hinged lid has to be opened/closed, a tube has to be placed in or picked out of the cage (which may have to be rotated into a proper position first), and buttons have to be pressed,
- the CEDEX cell counter, where a small tube has to be inserted or picked out of a carousel with very low clearance,
- a freezer, where a sliding lid has to be opened/closed and a tube placed into it,
- a barcode scanner, in front of which a barcode-labeled tube has to be held and possibly moved a little bit until the scanner has read the barcode, and
- several storage racks, from which tubes have to be picked.

A sequence of functions that meets the specific biotechnological requirements for sample management can then be issued by the process control system.

D. State Machine. The sequencing of high-level functions (actions) is verified by a state machine using attributes to describe the system state. This state machine enforces checks that ensure that no damage is done to the system in case of accidental mixing up of the command order. The actions can be divided into two types: (1) actions that end with the robot staying in kinematic contact with a device (to fixate a tube) and (2) actions during the execution of which there is no contact.

For the actions ending in contact with a device ("hold"), only the action that removes the contact ("take") is allowed as successor to prevent damaging the device. All other actions are only allowed if the robot is not in contact with any device. This important case is handled with the *holding* attribute. More restrictions are imposed by means of other attributes.

These restrictions are formulated by a list of *required* attribute values as preconditions of a command and a list of *changed* attributes as a result of its execution. Again, these lists are stored as ASCII text in the central database for easy maintenance. Table 2 shows the sequence of commands and their constraints used for our sample management.



(a) overexposed (b) normal (c) dark (d) very dark

Figure 13. Vision performance under different illumination brightness and chromacity caused by daylight illumination.

Table 1. Excerpt of Script Commands

command	arguments	explanation
checkstate	[REQUIRES ⟨list⟩] [CHANGES ⟨list⟩]	enforce safety and reasonability checks on sequences of high-level functions
call	⟨name⟩	invoke sub-script *name*
callif	TRANS|ROT ⟨trsf⟩ ⟨op⟩ ⟨limit⟩ ⟨name⟩	invoke sub-script *name* conditionally
open		open the gripper
close		close the gripper
pushspeed	⟨scale⟩	push the current speed on a stack and set the new speed as current speed times *scale*
popspeed		restore the previous speed from the stack
move	⟨poseq⟩	move the arm in Cartesian space according to a position equation
movej	⟨poseq⟩	move the arm in joint space according to a position equation
centerregion	⟨color⟩ ⟨trsf⟩ ⟨poseq⟩	center over the already centermost region of color *color* by changing *trsf* in *poseq*
confirmmodel	⟨device⟩ ⟨trsf⟩	confirm model *device* by checking all models and set *trsf* to its displacement
centermodel	⟨device⟩ ⟨trsf⟩ ⟨poseq⟩	center on model *device* by successively modifying *trsf* in *poseq*
fmove	[CTRL, ⟨spec⟩] [ABORT, ⟨spec⟩] ⟨poseq⟩	move in Cartesian space according to a position equation while obeying force constraints and/or limits
selectslot	⟨device⟩ ⟨flags⟩ ⟨trsf⟩	set *trsf* to the displacement of a free/full slot from *device* according to *flags*
settrsf	⟨trsf⟩ [⟨coordspec⟩|⟨trsf2⟩]	set *trsf*
newtrsf	⟨trsf⟩ ⟨coordspec⟩	create and set a new *trsf* to be visible until the current script is left
multtrsf	⟨trsf⟩ ⟨trsf2⟩	multiply *trsf* by *trsf2*
circle	⟨trsf⟩ ⟨trsf2⟩ ⟨trsf3⟩ ⟨poseq⟩	move the arm in Cartesian space according to a circular motion relative to the current position
mobile	move ⟨device⟩	move the mobile platform to *device*
mobile	forward ⟨distance⟩	move the mobile platform forward by *distance* meters (may be negative)
arm	start	start the arm by disabling brakes
arm	stop	stop the arm by enabling brakes
arm	approach ⟨poseq⟩	sequence of motions to unfold the arm from its park position into an optimal position (in terms of best joint scope) to approach *poseq*
arm	retreat ⟨poseq⟩	retreat from *poseq* and go into park position by applying the reverse order of commands as in "approach"

Table 2. State Machine Conditions[a]

	command	requires	changes
1	UnparkCharger	holding = charger	holding = false
2	PickCedexCedex	holding = false, gripper = empty	gripper = cedex
3	PlaceCedexWaste	holding = false, *gripper = cedex*	gripper = empty
4	PickTubeStorage	holding = false, gripper = empty	gripper = nunc, tubeempty = true, *barcode = false*
5	HoldTubeSampler	holding = false, gripper = nunc, tubeempty = true	holding = sampler
6	TakeTubeSampler	holding = sampler	holding = false, tubeempty = false
7	HoldTubePipet	holding = false, gripper = nunc	holding = pipet
8	TakeTubePipet	holding = pipet	holding = false
9	LoadAndRunCentrifuge	holding = false, gripper = nunc, centrifugeloaded = false	gripper = empty, centrifugeloaded = true
10	PickCedexStorage	holding = false, gripper = empty	gripper = cedex, tubeempty = true, *barcode = false*
11	HoldCedexPipet	holding = false, gripper = cedex	holding = pipet
12	TakeCedexPipet	holding = pipet	holding = false
13	PlaceCedexCedex	holding = false, *gripper = cedex*	
14	OpenFridge	holding = false, gripper = empty, fridge = closed	fridge = open
15	StopAndUnloadCentrifuge	holding = false, gripper = empty, centrifugeloaded = true	gripper = nunc, centrifugeloaded = false
16	HoldTubePipet	holding = false, gripper = nunc	holding = pipet
17	TakeTubePipet	holding = pipet	holding = false
18	PlaceTubeWaste	holding = false, *gripper = nunc*	gripper = empty
19	PickTubeStorageBarcode	holding = false, gripper = empty	gripper = nunc, tubeempty = true, *barcode = true*
20	HoldTubePipet	holding = false, gripper = nunc	holding = pipet
21	TakeTubePipet	holding = pipet	holding = false
22	HoldTubeScanner	holding = false, gripper = nunc, *barcode = true*	holding = scanner
23	TakeTubeScanner	holding = scanner	holding = false
24	PlaceTubeAndCloseFridge	holding = false, gripper = nunc, fridge = open	gripper = empty, fridge = closed
25	ParkCharger	holding = false	holding = charger

[a] Attributes that are optional and only used for reasonability are written in *italics.*

This table not only expresses safety constraints but also contains some "reasonability" constraints. It cannot cover *all* such constraints because the robot cannot fully observe the system state space. State information such as whether a tube is centrifuged could only be derived from the sequence of past actions but not actively verified with the given sensors. It is therefore deliberately discarded. This means that the resulting state machine

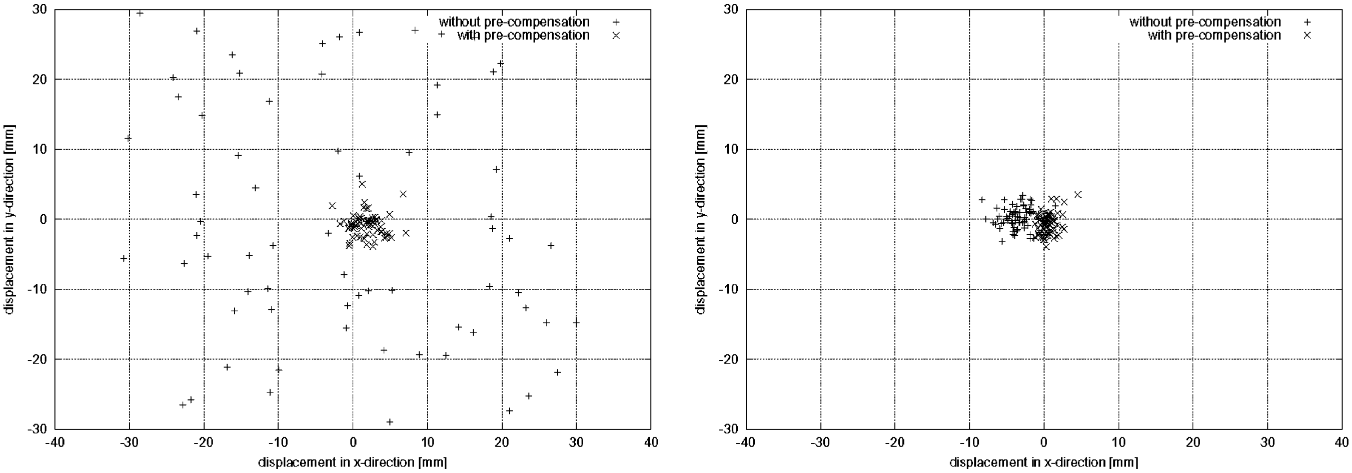


Figure 14. Positioning displacements as seen by the vision system. For the left graph the navigation was deliberately disturbed by Gaussian noise of up to 30 mm to show that large deviations are compensated before the vision system is invoked, whereas the right graph shows the normal situation.

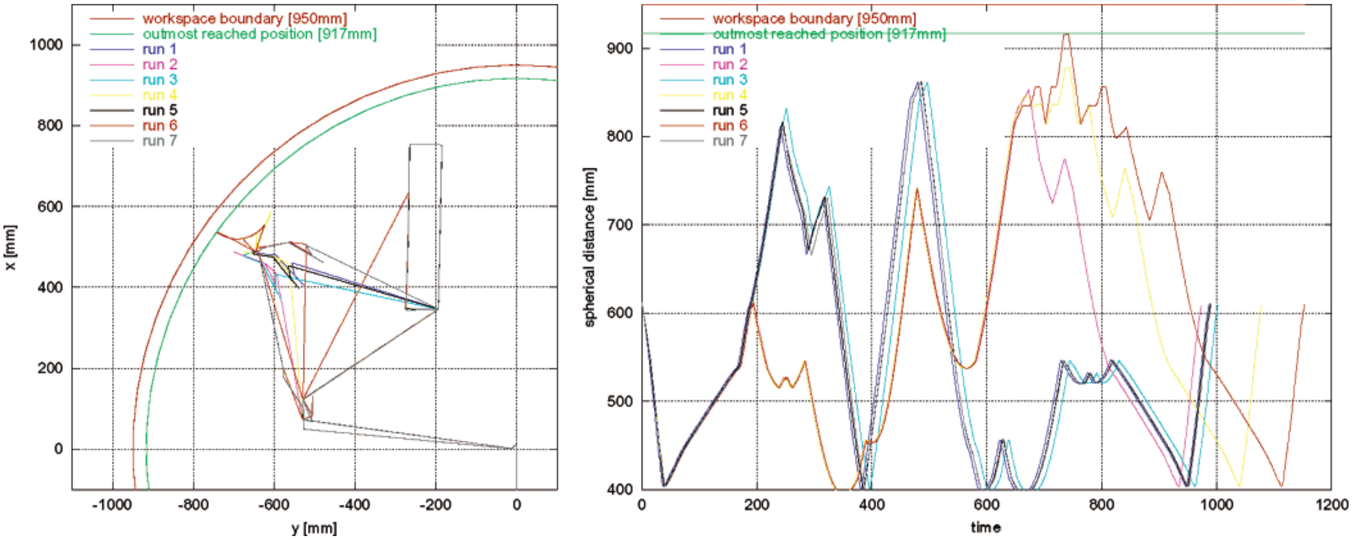


Figure 15. Workspace usage during several runs of operating the centrifuge. Both the two-dimensional projection of the workspace (left) and scalar distance from the origin (right) show that there is enough margin (roughly 3 cm) to ensure operability. Optimizations to gain more margin may be applied.

is nondeterministic, allowing multiple successive states for the same action. It is up to the user to impose more restrictions if desired.

Results of the System in Operation

The success of the manipulations depends mainly on the positioning accuracy of the mobile platform. The hand camera must initially be able to see the marker, and none of the subsequent operations must be out-of-reach for the arm.

Manual offline measurements have shown that the mobile platform's deviation from the goal position is typically less than 1 cm (regardless of the length of the path), a value not reached by many other platforms.

However, for lack of a global reference, this accuracy cannot be computed online. What can be computed online are the displacements seen by the vision subsystem when the robot arm has already compensated what the mobile platform has reported as error. Figure 14 shows an example of these displacements. Considering that the camera is equipped with a wide-angle lens and covers an area of ca. 10.5 × 8 cm at a viewing height of 14 cm, these displacements are well within bounds.

The question whether a motion is becoming out-of-reach with an increased platform positioning error can be answered by looking at how close to the outer workspace limit the arm usually comes. Figure 15 shows that for the centrifuge there is still a safety margin (the centrifuge is by far the most demanding of our devices in this respect).

Ahead-of-execution simulation might be used to detect if this margin becomes too low. In this case, the physical approach of the platform to the device could be repeated to cancel out noise or the stored position could be updated to cancel out more systematic influences.

We conclude this section by presenting a few typical pictures taken during a complete sampling cycle. Figure 16 shows the mobile manipulator as it approaches the filling station. It puts the sampling vial into one of the holes in the rack, then regrasps it to fixate it while it is being filled, and takes it away once it is completely filled up. The filling station is a simple device, which makes it possible to fill samples without any danger of contamination. It can be used both in automatic mode and in manual mode. It is basically a small and simple Car-

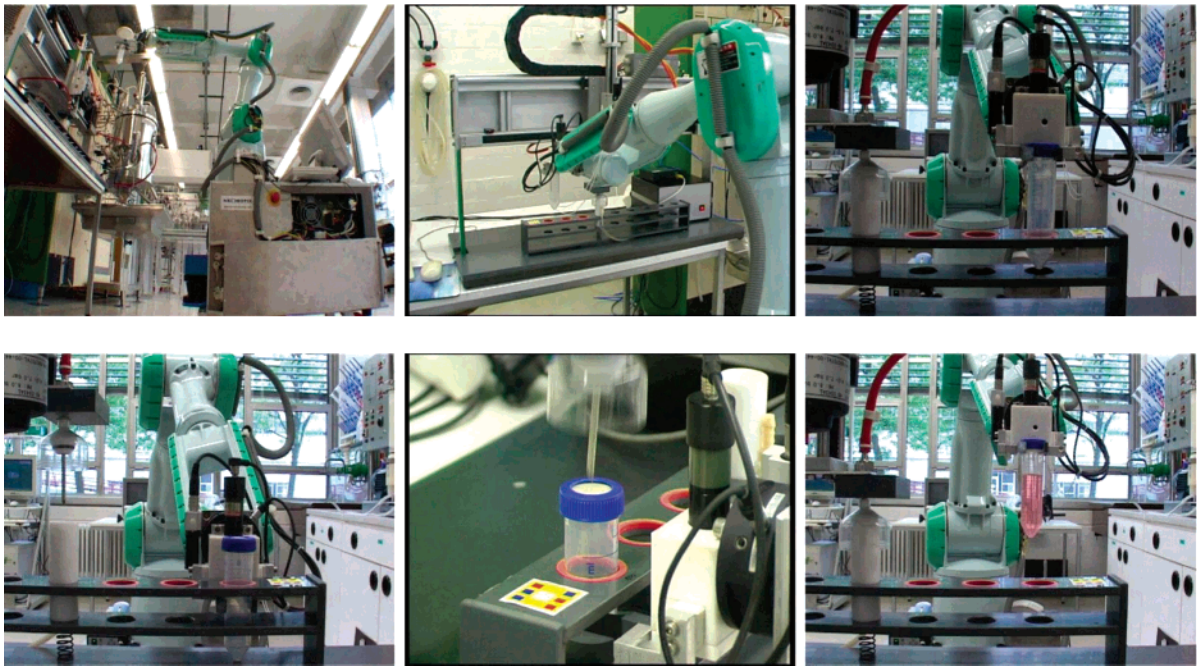


Figure 16. Robot and sampling station. Image sequence (from left to right and from top to bottom) showing the robot as it (i) approaches the sampling station, (ii) visually inspects and verifies the orientation of the station and the rack, (iii) puts the vial into the rack under force control, (iv) holds firmly to the vial so as to fixate it in its hole, (v) commands the sampling station to move the needle through the vial septum and to start the filling, and (vi) removes the vial with the cell culture fluid filled in.

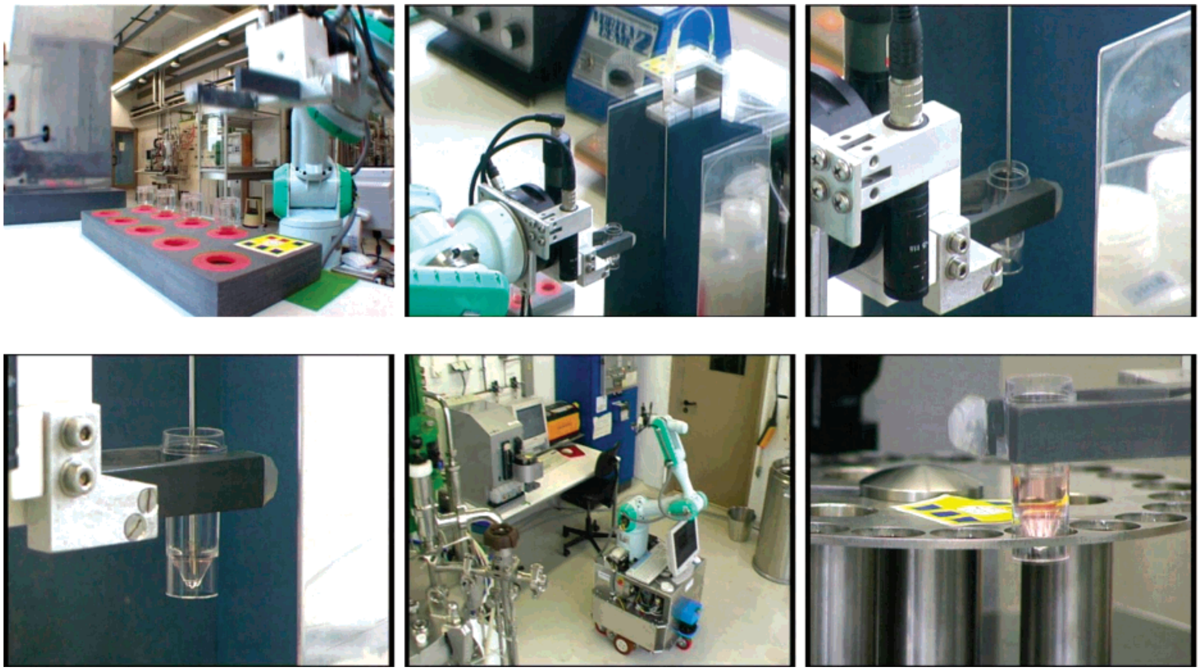


Figure 17. Feeding the CEDEX cell-counter. Image sequence (from left to right and from top to bottom) showing the robot (i) grasping a small CEDEX-vial, (ii) approaching the needle of the pipetting station, (iii) aligning the needle carefully in the small vial, (iv) commanding the pipetting station to fill in a small amount of cell culture fluid, (v) approaching the CEDEX cellcounter, and (vi) putting the vial into the CEDEX carousel with extremely high positioning accuracy.

tesian robot that moves a steam-sterilizable needle into the vial, which is held in a small rack.

The image sequence of Figure 17 shows the robot handling a small and open plastic vial that is used to transport small amounts of cell culture fluid from the pipetting station to the CEDEX cell counter. Not only is it smaller than the vial used for taking the sample (Figure 16), its shape is also quite difficult to handle. Note that the positioning accuracy of the visual servoing controller of the robot arm must be extremely high so as to ensure that the needle of the pipetting station is well-centered and also to guarantee that the insertion into the CEDEX carousel proceeds without friction between the vial and the carousel's holes.

Figure 18 shows an image sequence of the loading/unloading task of the centrifuge. Here it is important that the robot precisely determine both the location and the orientation of the centrifuge cabinet and also the rotor's orientation before pushing the vial into one of the centrifuge's vial holders. This is because the robot arm cannot reach the holder in all orientations of the turntable. It is hence particularly important to be able to

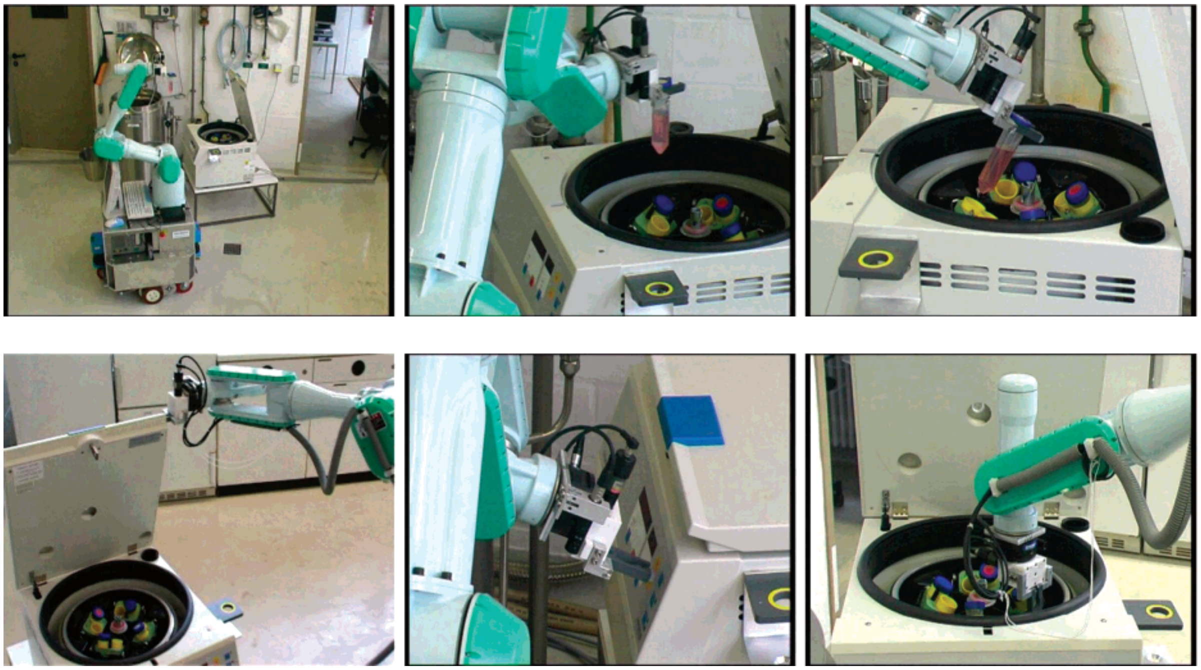


Figure 18. Operating the centrifuge. Image sequence (from left to right and from top to bottom) showing the robot (i) approaching the centrifuge and determining its position and orientation, (ii) analyzing visually the orientation of the turntable in the centifuge, (iii) putting the vial into one of the vial holders of the centrifuge, (iv) closing the cover of the centrifuge without any further help or tool, (v) pushing the buttons of the centrifuge to start its operation, and (vi) slightly rotating the turntable of the centrifuge to enable a safe retraction of the vial after having stopped the centrifuge and let it come to a standstill.

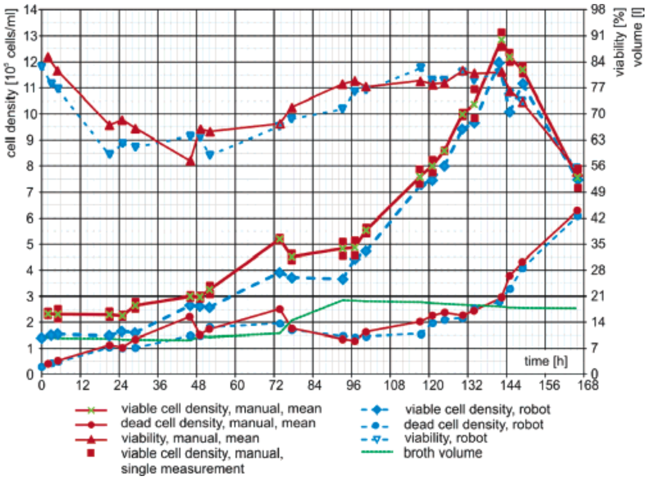


Figure 19. Measured parameters of a culture of hybridoma cells with a cultivation time of 168 h, comparing manual with automatic sampling.

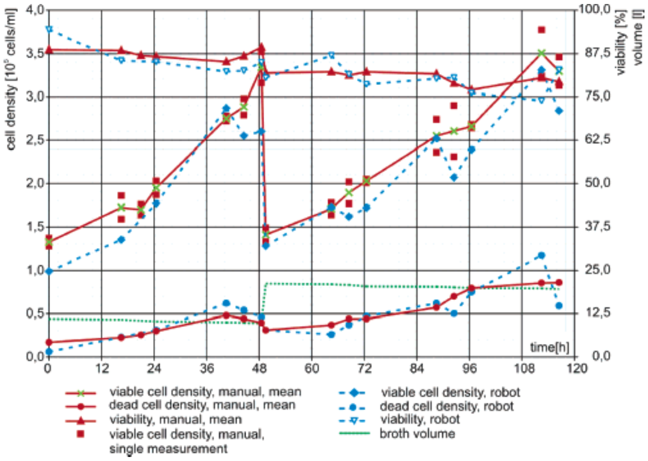


Figure 20. Measured parameters of a culture of recobinant CHO cells with a cultivation time of 120 h, comparing manual with automatic sampling.

adjust this position before inserting/retracting the vial from the centrifuge. The robot can do so by using a “rubber finger”, which it pushes on the rotor before gently turning it into a position of which it knows that the holder can be reached. Note that the robot is capable of closing the cover without any tools. It can also push all the buttons of the centrifuge, like a human operator would.

Finally, Figures 19 and 20 show the measurement results obtained manually and automatically from two real cultivations carried out in our laboratory over several days. In both modes, our CEDEX was used for cell counting. In manual mode, two samples were taken in parallel from the bioreactor to the CEDEX by the human operator, and then the mean value of the two results for the respective parameter was computed and plotted. In automatic mode, the robot took only *one* sample from the fermenter to the CEDEX. This was done to give a clear and unambiguous impression of the size of the errors that can be expected with our current setup (with respect to manual operation as the “true” reference). All measurements that were taken are plotted, including all outliers. The (small but visible) discrepancies between the curves showing the manual sampling and those of the automatic sampling procedures are mainly due to the construction of the pipetting station, whose cell culture fluid “storage volume” allows the cells to sediment between two samplings. This will be carefully redesigned in the future to minimize such differences. Apart from this systematic error, however, the reliability of the system and the general accuracy of the measurements have turned out to be more than satisfactory.

Conclusions

We have shown that a mobile robot system using unmodified standard laboratory equipment can be used to automate a complex technological process usually requiring human personnel (*18*). The robot system has so far been used in three supervised test cultivations without problems and is scheduled for unsupervised cultivations.

The presented methods provide the necessary accuracy to allow a robot with only limited sensory capabilities to safely operate a wide range of biotechnological devices.

In a straightforward next step, the culture parameters gathered by the system will be used to close the control loop and optimize a completely automated cultivation. Moreover, the system can be easily adapted to devices with a structure similar to those used in our setup. For example, a centrifuge with a different layout of buttons would only require teaching a new model and changing a few motion primitives. It is also easily possible to add *completely new devices* into the (sampling) process by combining the existing script commands with easy-to-write extensions. The system is therefore not limited to biotechnological laboratories or processes but can be used in a much wider area of similar situations.

Although the extension of the process using standard biotechnological equipment is simple as long as the robot need not be modified (in which case only new scripts, i.e., software, must be added), the possibility of constructing further "robot aides" (dedicated adapters) with associated scripts adds virtually unlimited possibilites for automating processes.

Finally, the potential of complete elimination of human intervention through high-fidelity telepresence and teleaction techniques may pave the way not only for remote-controlled plants but also for automatic plant maintenance with these highly autonomous robot systems.

Acknowledgment

This research has been funded by the German Federal Ministry of Economics and Labor (BMWI) through the Otto von Guericke e.V. as projects "AiF 11736 N/1" and "AiF 13223 N/1" in cooperation with Sartorius BBI Systems (formerly B. Braun Biotech), Melsungen (Germany), and Innovatis AG, Bielefeld (Germany). We would like to thank Bayer HealthCare, Berkeley (U.S.A.) for substantial financial support.

References and Notes

(1) Lloyd, J. E.; Hayward, V. *Multi RCCL User's Guide;* McGill University: Montréal, Québeq, Canada, 1989. http://www.cs.ubc.ca/spider/lloyd/rccl.html.

(2) Lloyd, J. E. *Trajectory Generation as a Nonlinear Filter*; Technical Report TR-98-11, Computer Science Department, University of British Columbia: Vancouver BC, Canada, 1998. http://www.cs.ubc.ca/cgi-bin/tr/1998/TR-98-15.pdf.

(3) Schneider, A.; Westhoff, D. Autonomous navigation and control of a mobile robot in a cell culture laboratory. Diploma Thesis, Faculty of Technology, University of Bielefeld, POB 100131, D-33501 Bielefeld, Germany, 2002.

(4) Mitsubishi Heavy Industries. *PA-10 General Purpose Robot Arm;* http://www.sdia.or.jp/mhikobe-e/products/mechatronic/e_index.html.

(5) Neobotix, c/o GPS GmbH, Department Robotics, Nobelstr. 12, D-70569 Stuttgart, Germany. http://www.neobotix.de

(6) Smith, S. M.; Brady, J. M. *SUSAN*−*A New Approach to Low Level Image Processing*, Technical Report TR95SMS1c; University of Oxford: Oxford, 1995.

(7) Gutmann, J.-S.; Burgard, W.; Fox, D.; Konolige, K. *An Experimental Comparison of Localization Methods*; International Conference on Intelligent Robots and Systems (IROS'98), Victoria, Canada, 1998.

(8) Gutmann, J.-S.; Fox, D. *An Experimental Comparison of Localization Methods Continued*; Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'02); Lausanne, Switzerland, 2002.

(9) Schmidt, S. Computational techniques in Kalman filtering. In *Theory and Applications of Kalman Filtering*; AGARDograph 139, NATO Advisory Group for Aerospace Research and Development: London, 1970.

(10) Latombe, J. C. *Robot Motion Planning;* Kluwer international series in engineering and computer science (4th print); Kluwer: Dordrecht, Boston, 1996.

(11) Innovatis AG. *The Automation of the Manual Trypan Blue Dye Method;* http://www.innovatis.com/download/Cedex_brochure.pdf.

(12) Nilsson, N. J. *Principles of Artificial Intelligence*; Springer-Verlag: New York, 1986.

(13) Hawerkamp, A. *Automated Trypan Blue Method for Optimal Cell Viability Determination*; American Biotechnology Laboratory: Feb. 2001.

(14) Zhang, J.; Knoll, A.; Schmidt, R. A neuro-fuzzy control model for fine-positioning of manipulators. *J. Rob. Autonomous Syst.* **2000**, *32* (2), 101−113.

(15) Poynton, C. *The Color FAQ*; http://www.poynton.com/ColorFAQ.html.

(16) Poggendorf, I.; Lütkemeyer, D.; Lehmann, J. *Vollautomatische, sterilisierbare Probenabfüllung als erster Schritt zur Roboterisierung der Probenentnahme und des Probenmanagements bei Zellkultivierungen im Pilotbioreaktor,* Tagungshandbuch 4. Dresdner Sensor-Symposium; Dresden, 1999.

(17) Lütkemeyer, D.; Poggendorf, I.; Scherer, T.; Zhang, J.; Knoll, A.;Lehmann, J. First steps in robot automation of sampling and sample management during cultivations of mammalian cells in pilot scale. *Biotechnol. Prog.* **2000**, *16*, 822−828.

(18) A small WWW site will be available at http://leonardo.cs.tum.edu by the time of publication of this article. It will provide additional information about the system and videos showing the live operation of the robot.

Accepted for publication August 20, 2004.

BP049759V