A Service Robot for Automating the Sample Management in Biotechnological Cell Cultivations

Torsten Scherer[*], Iris Poggendorf[*], Axel Schneider[†], Daniel Westhoff[‡]
Jianwei Zhang[‡], Dirk Lütkemeyer[*], Jürgen Lehmann[*], Alois Knoll[§]
[*]: Dept. of Cell Culture Technology, Faculty of Technology, University of Bielefeld, POB 100131, 33501 Bielefeld, Germany
[†]: Dept. of Biological Cybernetics, Faculty of Biology, University of Bielefeld, POB 100131, 33501 Bielefeld, Germany
[‡]: Dept. TAMS, Faculty of Computer Science, University of Hamburg, Vogt-Kölln-Straße 30, 22527 Hamburg, Germany
[§]: Dept. of Computer Science, TU München, Boltzmannstraße 3, 85748 Garching, Germany
[*]: EMail: itschere@techfak.uni-bielefeld.de

***Abstract*— In this paper we present a mobile robot system that is capable of automating the sample management in a biotechnological laboratory. The system consists of a mobile platform and a robot arm. It can navigate freely in the laboratory and operate standard devices needed for the sample management. The platform uses an extended Kalman filter for localization and the A^* algorithm for path planning on a tangent graph computed from the laboratory's map. Motion execution has been designed to be as predictable as possible to not irritate, disturb or harm human personnel. The robot arm uses color vision to detect devices and compensate for positioning errors. The parameters and tasks needed to operate the devices are specified in simple scripts to allow quick and easy adaptations to other situations.**

I. Introduction

Sample management is an essential part of the biotechnological process of mammalian cell cultivation for the production of biopharmaceuticals. Its particular importance lies in monitoring the culture and determining the optimal harvest time. Steps of this process are taking samples from the bioreactor at regular intervals, counting the cells and determining their viability, separating cells from the broth using a centrifuge and storing an 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.

A normal cultivation of mammalian cells usually takes up to two 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. Since this includes both nights and weekends it presents a high cost factor. Human personnel also introduces unpredictable errors when judging the sample, depending on their training and fatigue.

Attempts to achieve an automatic 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. Already the slightest change in the analytical process may render 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.

On the other hand, semi-automatic stand-alone analytical devices for most culture parameters already exist. For example the CEDEX cell counter [11] used in our setup automates the cell count using the standard trypan-blue method. It allows a fast and reliable analysis of the sample by using a computer vision system on a microscope to count and classify mammalian cells [12]. Yet, the CEDEX has to be loaded/unloaded and operated by human personnel.

To overcome these disadvantages we have built a mobile robot system that automates the *entire* sampling process, while also making it more reliable and consistent [16]. The system consists of a mobile platform that navigates freely in the laboratory and a robot arm to carry the sample and operate devices. It presents a new approach because it uses standard laboratory equipment with only the least necessary amount of modifications. The intention is to explicitly use existing devices and allow human personnel to use them for other tasks while the robot is idle, and thus eliminate the need to have 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. Details about the sampling routine and the devices used are given in [15].

In this paper we present the precise and reliable mechanism for localization and navigation, as well as the color vision system to classify devices and detect and compensate for positioning errors. It will be shown that several of the implementation details have been designed to allow easy alteration of system parameters by non-expert personnel. Results demonstrating the system's reliability during a test cultivation will be presented.

II. System Description

The system consists of a Mitsubishi PA-10 robot arm [4] mounted on a mobile platform [5] as in Fig. 1. The platform is equipped with a differential drive with odometers, a gyro compass, two SICK LMS-200 laser range finders and a PC running Linux. The drive wheels and the gyro are connected to

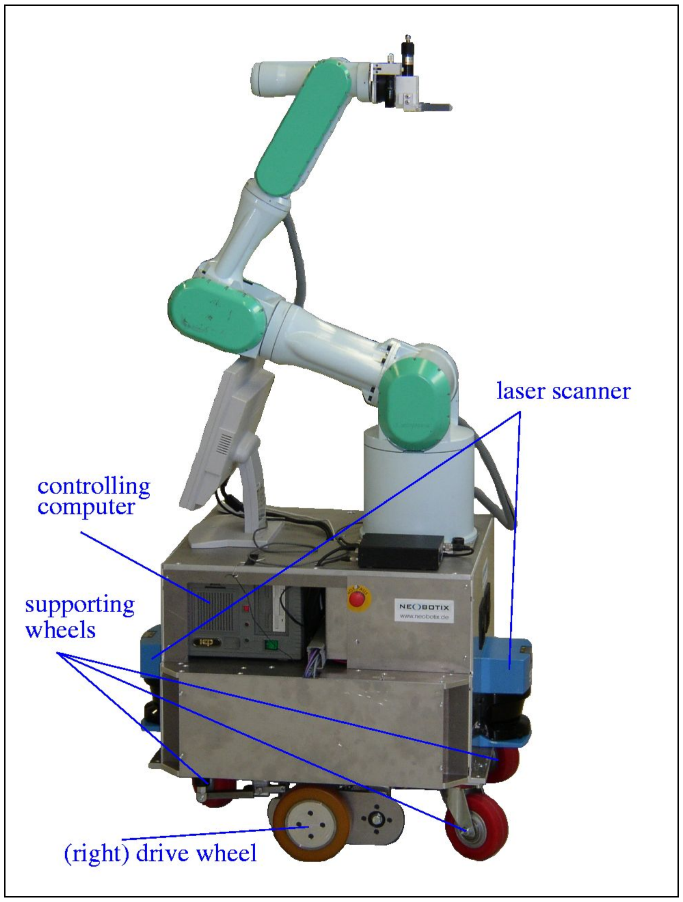


Fig. 1. The mobile platform and robot arm.

the PC via CAN bus. The laser range finders are connected to the PC via high speed RS422 ports. They measure the distance to obstacles in a range of $180°$ each and detect special retro-reflecting marks used for localization.

The arm is equipped with a wrist-mounted force/torque sensor (FTS), a microhead color camera and an electric parallel yaw gripper as in Fig. 2. The arm is connected to the PC via an ARCNET network and controlled at joint controller level by a modified version of RCCL [1]. The FTS is connected to the PC via an ISA bus receiver board, the camera via a Matrox Meteor PCI framegrabber board and the gripper via two general purpose I/O bits on the FTS board.

III. Mobile Robot Control

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:

1) The exact global position and orientation of the platform have to be determined (the localization problem).
2) A path from the current robot position to a goal position has to be found (the navigation problem).
3) A control mechanism has to be implemented that moves the robot according to the computed path (the motion execution problem).

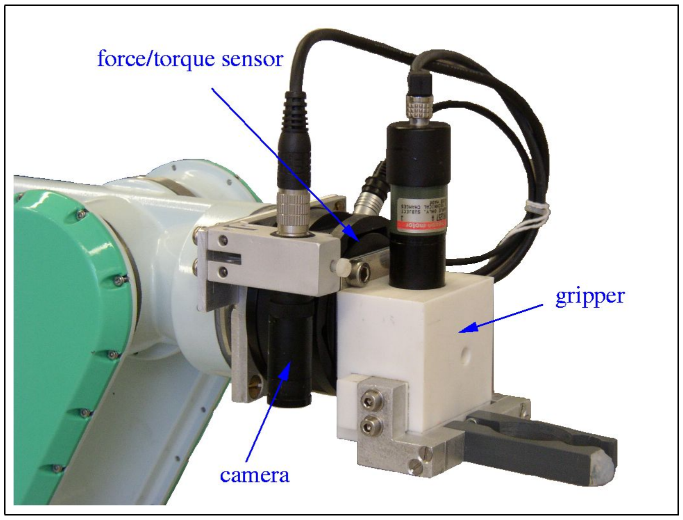


Fig. 2. The robot arm tool.

A. Localization

Experiments done by Gutmann et. al. in [7] and [8] show that Kalman filtering techniques yield the most precise results for solving the localization problem. In this work an extended Kalman filter (EKF) by Schmidt [9] is used. The system-state vector $\vec{x}_t$ contains the platform's position and velocity and the angular velocities of the drive wheels as well as the positions of the laser reflector marks. These reflector marks have to be distributed throughout the mobile robot's workspace and serve as landmarks with a known global position. The complete state vector $\vec{x}_t$ is:

$$\vec{x}_t = \left(\omega_R, \omega_L, \dot{x}, \dot{y}, \dot{\phi}, x, y, \phi, x_{f1}, y_{f1}, ..., x_{fi}, y_{fi}\right)_t^T .$$

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

$$\vec{z}_t = (\omega_R, \omega_L, \dot{\varphi}, d_{f_1}, \alpha_{f_1}, ..., d_{f_i}, \alpha_{f_i})_t^T .$$

It has to be emphasized that the EKF is able to merge sensor information of very different accuracy and give an estimate of past, present and future system states. In our system it provides a position estimate up to 38 times per second.

B. Navigation

In order to solve the navigation problem, the A^* algorithm and tangent graph discussed by Latombe in [10] have been applied for planning a path from the current position to the desired goal position at a device. The A^* algorithm searches for the shortest path on a tangent graph built on a set of polygons. These polygons represent the static obstacles in the mobile platform's workspace and are generated by expanding a simple vector representation of the laboratory's map.

In order 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 amount. Based on that, a tangent graph is that set of straight lines which connect all polygon vertices without going through any polygon body.

The A^* algorithm returns the combination of those tangents which form the shortest path from start to goal. An example can be seen in Fig. 3, showing a path from a point S to a point G.

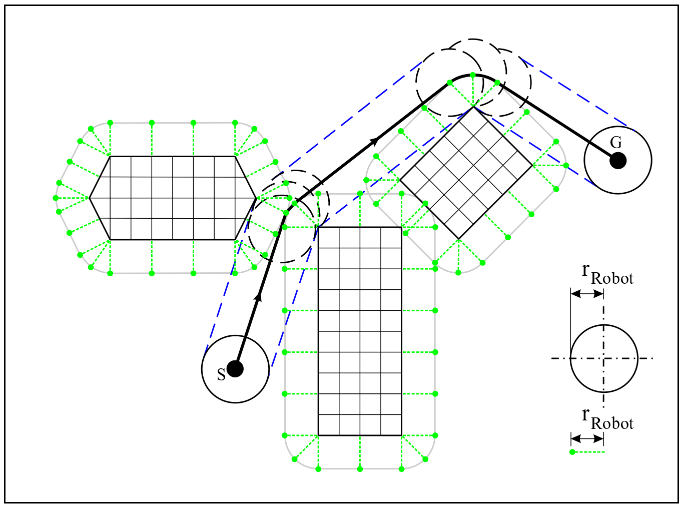


Fig. 3. Expanded map with path. The three obstacles are expanded by the radius of a circle which surrounds the robot.

C. Motion Execution

Motion execution does not utilize any behaviours (like "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 inflexible, but makes the platform predictable and verifiable - features which 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 towards the target with a $1d$-trapezoidal velocity profile [2]. The motion is brought to rest at a line perpendicular to the trajectory through the target point. This does allow 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 come within 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.

IV. 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 - for one device - only about one millimeter is an indispensable prerequisite for successful robot manipulations. A color vision system is therefore used to determine and compensate these errors.

A. Color Vision

Cameras provide the most natural and comprehensive information. However, they require a very large amount of processing to extract the essential information. It is therefore desirable - if not simply 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.

If you are looking at "color" technically the often used RGB color representation has the disadvantage that it mixes color and brightness information. The standardized CCIR-601 YUV color representation [14] separates the brightness (Y) from the color (U/V) information and is therefore better suited for our purpose. Another advantage of YUV is that it is the native SVHS video signal format and can therefore be processed by the framegrabber board 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-noise-ratio. Fig. 4 shows the Y, U and V channels of a sample image.

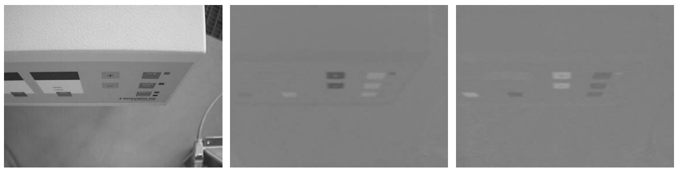


Fig. 4. Y, U and V channel (from left to right) of a CCIR-601 YUV color image (original images, not contrast-maximized).

One approach to detect colored regions is to search for their edges in the U/V images. Classic edge detection using a derivation-based approach like Sobel filters behaves poorly on signals with a low signal-noise-ratio. Other approaches like the SUSAN detector by Smith in [6] usually perform better in these cases. Since 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 which are found in the image. Fig. 5 shows 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 could also be used as input format.

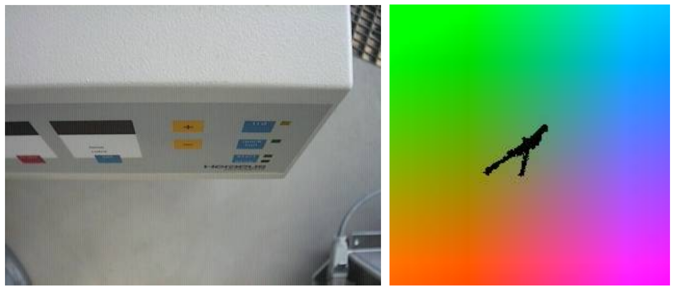


Fig. 5. Sample image of the centrifuge (left) and its colors used in the U/V plane (right, the colors in the background are indistinguishable in b/w print because all pixels have the same gray value). It can be seen that only a fraction of the full range of color saturation is used.

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.

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

In this rotated coordinate system a similarity measure

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

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 in accepting variance of the color saturation than variance of the color value, as can be seen in Fig. 6.

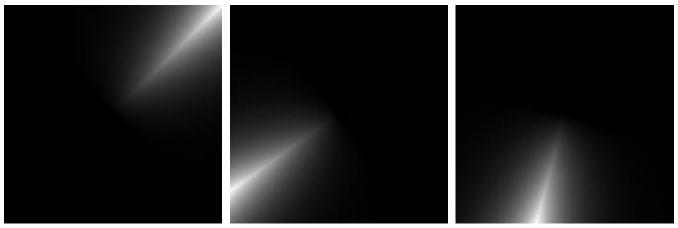


Fig. 6. Similarity measure of the colors "blue", "yellow" and "red" of the U/V space in Fig. 5.

The measure is then applied to the known colors of objects, yielding similarity "images" of colors as in Fig. 7. An example of how the situation with most brightness information removed looks like for the robot is given in Fig. 8. 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 thus determined are stored in a list.

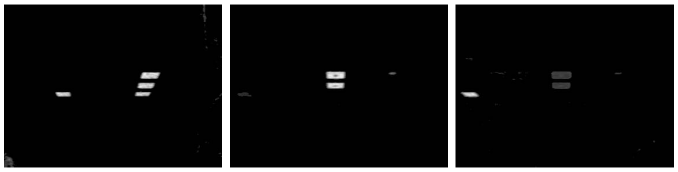


Fig. 7. Similarity of pixels to the colors "blue", "yellow" and "red" (from left to right) of the image in Fig. 5. The colors "yellow" and "red" are similar enough to still yield a low ranking for their counterparts.

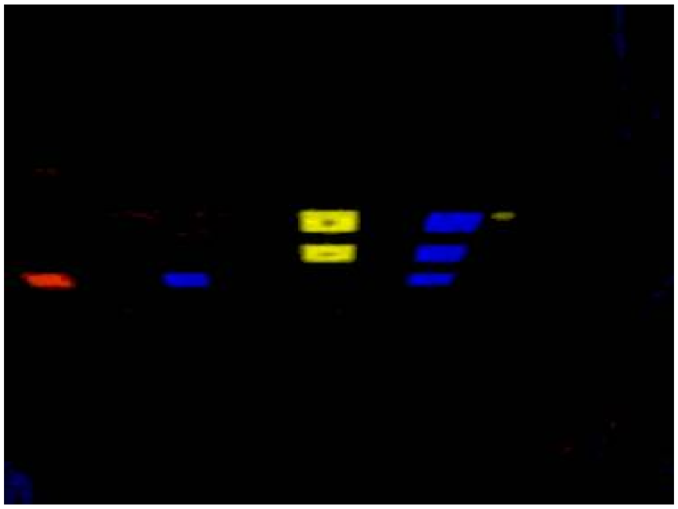


Fig. 8. Combined similarity image.

B. Model Matching

These regions are 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 like neural networks or fuzzy controllers like [13] this model-based approach needs only one training image for the one desired situation. 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). Learning-based approaches usually only perform correctly if the learn space is homogeneously covered by learn data.

The matching is done by looking at an assignment of COG pixels. Allowing only the xy-translation and z-rotation of a $2d$-model, the relation between each image pixel $\vec{p}_i$ and its model pixel $\vec{p}_m$ can be described as

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

which can be rearranged to

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

Combining the equations of at least two pixels of a complete model yields an overdetermined equation system[1]

$$M_m \cdot \vec{u} \quad = \quad \vec{p}_i$$

which can be solved with the pseudo inverse

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

[1]We take $\sin\alpha$ and $\cos\alpha$ as linearly independent for simplification.

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

$$e = \|M_m \cdot \vec{u} - \vec{p}_i\|$$

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

As can be seen in Fig. 9, this approach does not deal with perspective effects caused by displacements and/or lens errors. A $3d$-model to compensate these effects has been tested,

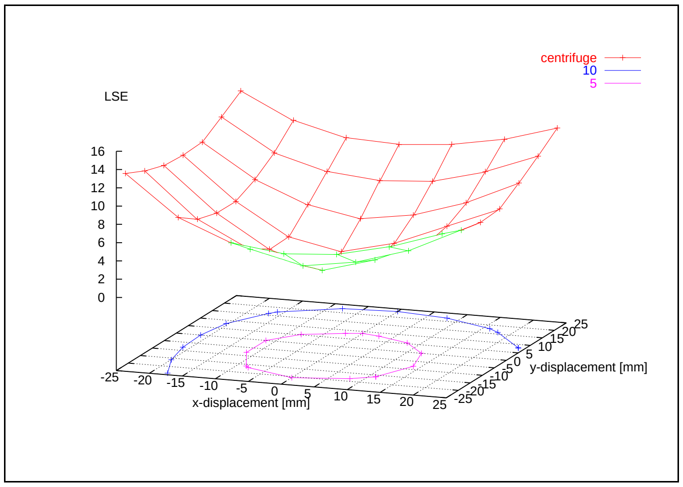


Fig. 9. LSE changes caused by perspective effects at different displacements. The circles labelled “5” and “10” show isobars of the error function.

but has been found to be less well conditioned. Despite the systematic error thus introduced, the $2d$-model still allows safe classification of devices (see Fig. 10).

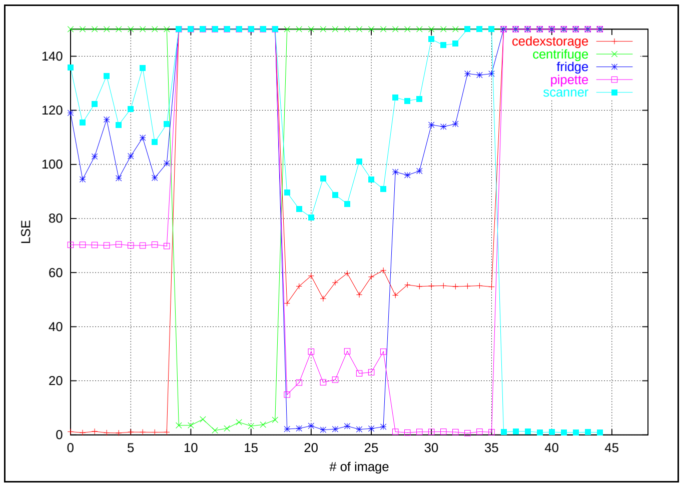


Fig. 10. LSE allows an unambiguous classification of different devices with different displacements. Error values have been clipped at 150.

C. Illumination Invariance

Since the vision system uses no brightness information it is very tolerant against changes in the illumination. Fig. 11 shows that the amount of noise in the color data does increase and the quality of the resulting regions does decrease 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 the classification will fail.

V. 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 like “*fetch a tube*”, “*hold a tube under the pipette*” or “*place a tube in the centrifuge and start it*”. To allow easy adaptation of these functions a couple of simplifying features have been incorporated.

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 re-read each time the function is invoked. Since they are stored as ASCII text they can be trivially changed.

It has proven to be impossible to use an existing script language/interpreter (like tcl/tk, perl, python etc.) and 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 OS and only a rather trivial and custom script language. This lack of complexity in the script language in turns allows people with comparably little training to do changes.

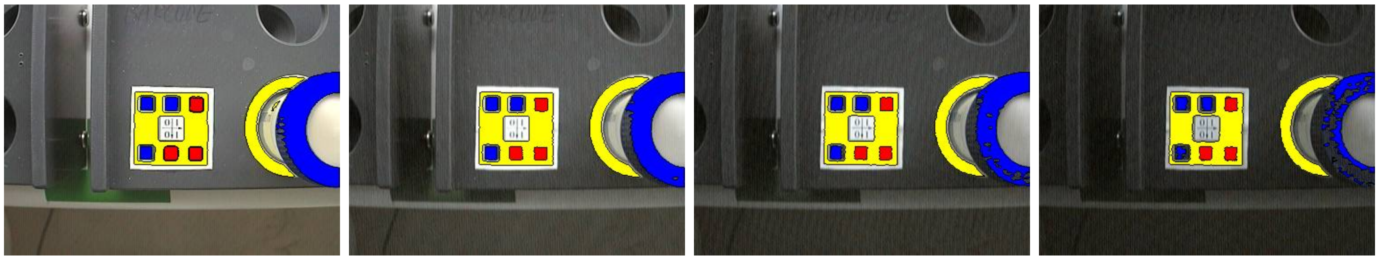
B. Script Commands

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 I.

C. High-Level Functions

With these script commands the set of high-level functions is realized. These functions can operate

- 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, where a small tube has to be inserted or picked out of a rotary disk 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-labelled 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) overexposed (b) normal (c) dark (d) very dark

Fig. 11. Vision performance under different illumination conditions.

TABLE I

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 ”start”

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 high-level functions (actions) are secured by a state machine using attributes to describe the system state. This state machine enforces checks which 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 that do not.

For those actions that end staying in contact with a device (“hold”), only that action that removes this contact (“take”) is allowed as the next command to prevent damaging the device. All other actions are only allowed if the robot is not in contact with any device. This most 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 pre-conditions 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 II shows the sequence of commands and their constraints used for our sample management.

This table not only expresses safety, but also a few reasonability contraints. It cannot cover *all* reasonability constraints because the robot cannot fully observe the system state space. State information like whether a tube is centrifuged or not could only be derived from the sequence of past actions, but not actively verified with the given sensors. It is therefore

TABLE II

STATE MACHINE CONDITIONS. ATTRIBUTES THAT ARE OPTIONAL AND ONLY USED FOR REASONABILITY ARE WRITTEN IN *italics*.

	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	HoldTubePipette	holding=false, gripper=nunc	holding=pipette
8	TakeTubePipette	holding=pipette	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	HoldCedexPipette	holding=false, gripper=cedex	holding=pipette
12	TakeCedexPipette	holding=pipette	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	HoldTubePipette	holding=false, gripper=nunc	holding=pipette
17	TakeTubePipette	holding=pipette	holding=false
18	PlaceTubeWaste	holding=false, *gripper=nunc*	gripper=empty
19	PickTubeStorageBarcode	holding=false, gripper=empty	gripper=nunc, tubeempty=true, *barcode=true*
20	HoldTubePipette	holding=false, gripper=nunc	holding=pipette
21	TakeTubePipette	holding=pipette	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

deliberately discarded. This means that the resulting state machine is non-deterministic, allowing multiple successive states with the same action. It is up to the user to impose more restrictions until the point where e.g. exactly only the sequence in table II is allowed.

VI. RESULTS

The success of the manipulations depends mainly on the positioning accuracy of the mobile platform. Manual offline measurements have shown that the mobile platform's deviation from the goal position is typically less than $1\,cm$ - a value not reached by many other platforms. However, failing 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. Fig. 12 shows an example of these displacements. Considering that the camera is equipped with a wide-angle lens and covers an area of app. $10.5 \times 8\,cm$ at a viewing height of $14\,cm$ these displacements are well within bounds.

Another aspect is whether compensation of positioning errors results in a motion target being out-of-range for the arm. Fig. 13 shows that for the centrifuge there is still a (reasonable) safety margin, though the centrifuge is most demanding in this respect. Ahead-of-execution simulation might be used to detect if this margin becomes too low. In this case, the platform's approach to the device could be repeated to cancel out noise or the stored position could be updated to cancel out more systematic influences.

VII. CONCLUSIONS

We have shown that a mobile robot system using largely unmodified standard laboratory equipment can be used to automate a complex technological process usually requiring human personnel. The presented methods provide the necessary accuracy to allow a robot with only limited sensoric capabilities to safely operate a wide range of biotechnological devices.

The system can be very easily adapted to devices with a similar structure as 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.

Even completely new devices could be incorporated rather easily if they can be operated with the existing script commands. The system is therefore not at all limited to biotechnological labs or processes, but can be used in a much wider area of similar situations.

Finally, in a next step of automation, the culture parameters gathered by the system could be used to close the control loop and optimize a completely automatated cultivation.

ACKNOWLEDGEMENTS

This research has been funded by the German Federal Ministry of Economics and Labour (BMWI) through the Otto von Guericke e.V. as projects “AiF 11736 N/1” and “AiF 13223 N/1” in cooperation with B. Braun Biotech, Melsungen (Germany) and Innovatis AG, Bielefeld (Germany). We also like to thank Bayer HealthCare, Berkeley (USA) for financial support.

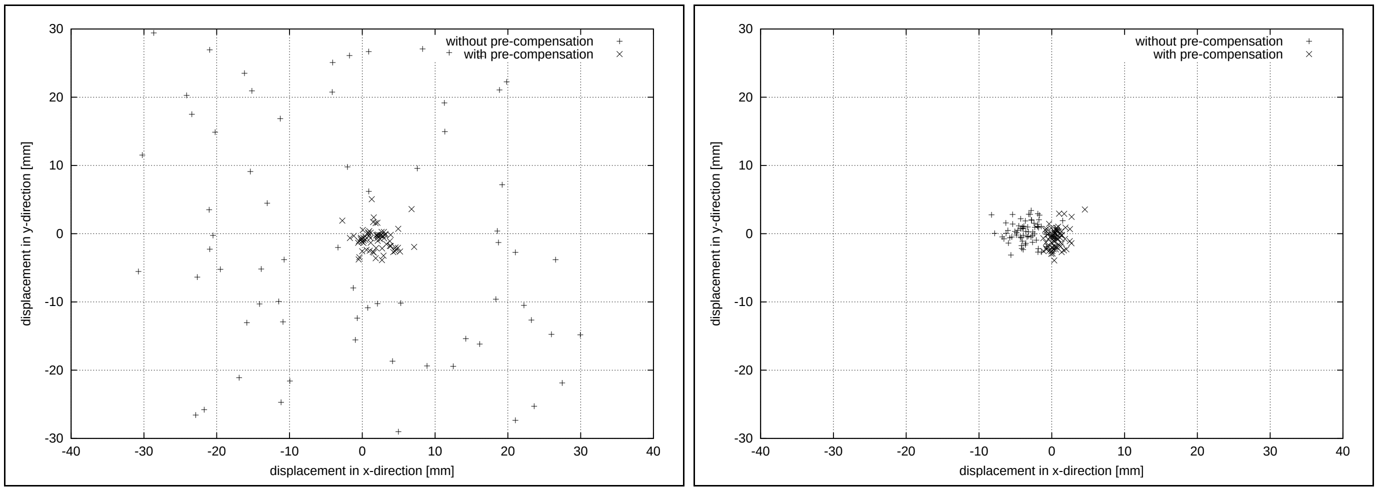


Fig. 12. Positioning displacements as seen by vision system. For the left graph the navigation was deliberately disturbed by gaussian noise of up to $30\,mm$ to show that large deviations are compensated before the vision system is invoked, while the right graph shows the normal situation.

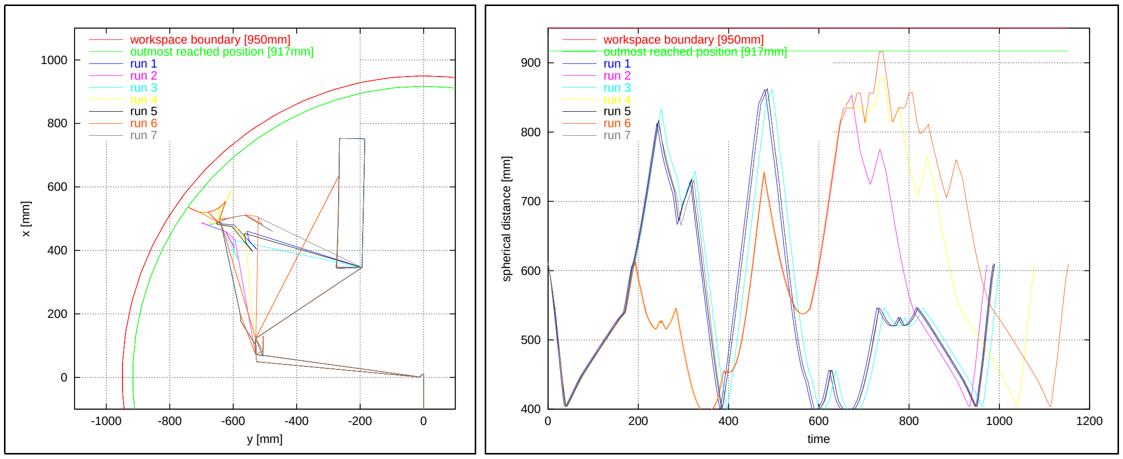


Fig. 13. Workspace usage during several runs of operating the centrifuge.

References

[1] John E. Lloyd and Vincent Hayward (1989), *"Multi RCCL User's Guide"*, McGill University, Montréal, Québeq, Canada. http://www.cs.ubc.ca/spider/lloyd/rccl.html

[2] John E. Lloyd (1998), *"Trajectory Generation as a non-linear Filter"*, Technical Report TR-98-11, Computer Science Department, University of British Columbia, Vancouver B.C., Canada. http://www.cs.ubc.ca/cgi-bin/tr/1998/TR-98-15.pdf

[3] Axel Schneider and Daniel Westhoff (2002), *"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.

[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] S. M. Smith and J. M. Brady (1995), *"SUSAN - A new approach to low level image processing"*, Technical Report TR95SMS1c, University of Oxford.

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

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

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

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

[11] Innovatis AG, *"The automation of the manual Trypan blue dye method"*. http://www.innovatis.com/download/Cedex_brochure.pdf

[12] A. Hawerkamp (2001), *"Automated Trypan Blue Method for Optimal Cell Viability Determination"*, American Biotechnology Laboratory Feb. 2001.

[13] J. Zhang, A. Knoll and R. Schmidt (2000), *"A Neuro-Fuzzy Control Model for Fine-Positioning of Manipulators"*, Journal of Robotics and Autonomous Systems 32 (2-2), 101-113.

[14] C. Poynton, *"The Color FAQ"*. http://www.poynton.com/ColorFAQ.html

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

[16] D. Lütkemeyer, I. Poggendorf, T. Scherer, J. Zhang, A. Knoll and J. Lehmann (2000), *"First Steps in Robot Automation of Sampling and Sample Management during Cultivations of Mammalian Cells in Pilot Scale"*, Biotechnology Progress, 2000, Vol. 16, T. 5, 822-828, American Chemical Society.