Design and Control of an Intelligent Dual-Arm Manipulator for Fault-Recovery in a Production Scenario

Jose de Gea, Johannes Lemburg, Thomas M. Roehr,
Malte Wirkus, Iliya Gurov and Frank Kirchner
DFKI (German Research Center for Artificial Intelligence)
Robotics Innovation Center
Bremen, Germany

Abstract

This paper describes the design and control methodology used for the development of a dual-arm manipulator as well as its deployment in a production scenario. Multimodal and sensor-based manipulation strategies are used to guide the robot on its task to supervise and, when necessary, solve faulty situations in a production line. For that task the robot is equipped with two arms, aimed at providing the robot with total independence from the production line. In other words, no extra mechanical stoppers are mounted on the line to halt targeted objects, but the robot will employ both arms to (a) stop with one arm a carrier that holds an object to be inserted/replaced, and (b) use the second arm to handle such object. Besides, visual information from head and wrist-mounted cameras provide the robot with information such as the state of the production line, the unequivocal detection/recognition of the targeted objects, and the location of the target in order to guide the grasp.

1 Introduction

Industrial robots have been present in factories for almost 50 years, when in 1961 the Unimate robot was deployed at a General Motors plant, following the invention from George Devol in cooperation with Joseph Engelberger in 1956. Ever since, the use of robots has been steadily increasing, especially in automotive industry which accounts for almost 60% of the total robot sales [2]. But there are several potential markets in which robots have not yet been introduced, mainly due to the challenging requirements and the high costs involved. Nowadays, robots are mainly deployed on large-volume manufacturing scenarios, where tasks are very repetitive, and whose environments are strictly controlled. In this work though, we present the deployment of a robotic manipulator on a next-generation industrial automation facility, the $SmartFactory^{KL}$ [4], a factory whose components can be arbitrarily modified, and that autonomously reconfigures itself according to the current context. In this scenario, a robot should be able to deal with a mostly unknown, dynamically-changing environment as well as with variations on the properties and geometry of the goods to handle.

Such environmental challenges require additional sensor equipment, reactive and dynamical software, and novel manipulation concepts. These requirements shaped the specifications of our robotic system. In terms of sensor equipment, the ability to react to enviromental changes is primarily provided by visual information. The robot needs to be able to visually scan the environment and recognize the current context. A stereo camera mounted on the head of the robot provides information about the objects in the environment as well as about their position. A high-speed camera on the robot's wrist guides the arm towards the object to grasp. On the other hand, the robot is required to be independent from the production line, i.e. it cannot rely on extra equipment mounted on the line. The reason stems mainly from the fact that a fault on the line can appear anywhere and it is not practicable to mount extra sensors/actuators all over it. That requirement led to the development of a dual-arm system that will combine the use of both arms to solve complex tasks. To our knowledge, no other dual-arm robot exists in industrial applications except for the Motoman SDA10 [1]. However, our robot additionally includes vision and computing power on the same platform. Figure 1 shows our dual-arm robot manipulating objects from a simple $SmartFactory^{KL}$ module present at our laboratories.

2 Scenario Description

The robot was aimed to be deployed in a real production scenario, as part of the so-called $SmartFactory^{KL}$, a modular, and self-organising production factory. The goal was a dual-arm robot working in conjunction with the $SmartFactory^{KL}$ and helping recover the production line from a fault. In the current scenario, a fault is defined as a carrier having lost its pill container, a condition that is signaled by the module by illuminating an or-

978-1-4244-2728-4/09/$25.00 ©2009 IEEE

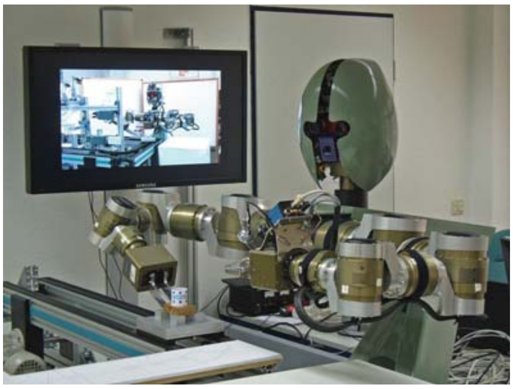


Figure 1. Dual-arm manipulator system

ange light. In that situation, the robot has to: (1) become aware of the fact that a pill container was lost; that is, the robot has to perceive the light signaling issued by the $SmartFactory^{KL}$ module, (2) recognise the empty carrier that triggered the faulty condition, (3) stop the empty carrier, without intervening on the line, i.e. without stopping the conveyor belt, (4) insert an 'emergency' pill container on the empty carrier, (5) let the 'emergency' pill container be filled by the $SmartFactory^{KL}$ module with the corresponding pills, (6) detect the 'emergency' pill container coming back, (7) remove the 'emergency' pill container from the line (the pills have to be removed from the system, as their recipient is unknown at that stage).

3 Mechanics Design

A metal (non-mobile) body sustains the two arms and keeps the power supplies for the robot, whereas most of the electronic equipment is kept on the robot's head (Fig. 2). The primary design constraint is the support of the visual sensors, a stereo, and a 3D-camera, which inevitably have to face the front with an unobstructed field of vision, as well as to be movable about the pitch and yaw axis. Besides all the constraints placed by functional reasons, an effortless maintenance is a major design aspect in order to achieve high accessibility of the system.

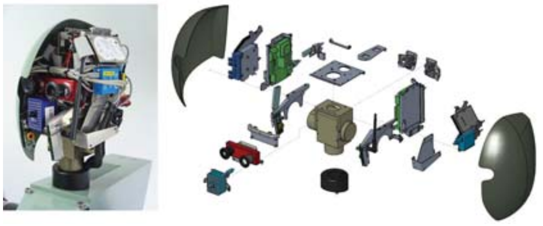


Figure 2. Explosion view of the head-components

4 Hardware

This section aims at describing the hardware components used in our robotic platform, as well as giving a brief statement about its purpose for the system. Figure 3 shows the main components of the system as well as the communication interfaces used between them.

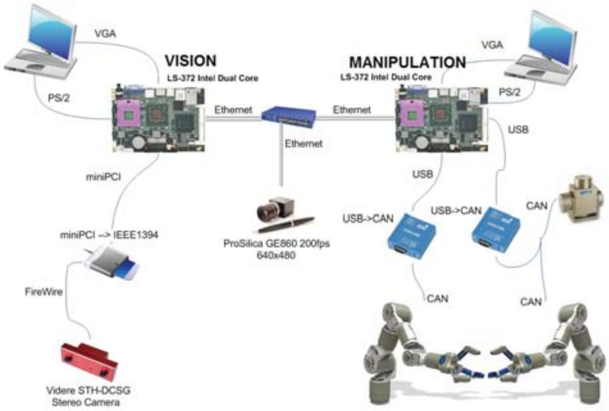


Figure 3. Hardware components

4.1 Control PCs

The *brain* of the robot system are two industrial 3.5" single-board computers (SBC) from COMMELL, model LS-372, with Intel Core 2 Duo Mobile T9300 processors at 2.5 GHz. Additionally, each board includes 1 GB RAM DDR2 memory, 1 Gigabit Ethernet interface, mini-PCI socket, two USB 2.0, two serial ports, and UltraATA33 IDE support for hard drives, among other interfaces.

- The *Manipulation Computer* is the main control board which controls the arms and requests, when necessary, camera information from the *Vision Computer* to guide the arm towards the objects.

- The *Vision Computer* is used for processing the data received from the two cameras: the stereo camera located on the head and the wrist camera. The former is used for object recognition, and the latter for visual servoing tasks.

4.2 Robot Arms

The dual-arm system is based on modular joints from Schunk. Each arm is composed of seven modules, mixing four different module sizes (PRL120, PRL100, PRL080 and PRL060), with peak output torques ranging from 10 Nm to 372 Nm.

The system uses two independent CAN bus lines, one line for controlling one arm plus the pan-tilt servo unit that controls the two degrees of the head, and the second line for controlling the second arm.

4.3 Cameras

The robot is equipped with a set of cameras which provides the robot with valuable information about its dynamically-changing environment, thus allowing it to react according to that information. A stereo camera is used for the recognition of the objects to manipulate. This stereo camera is equipped with two different camera lenses: the left lens provides a wide angle view used to obtain a view of the whole scene, whereas the right lens provides a high-resolution and detailed view of a small area where the objects to manipulate are expected to be found. A second camera is located on the robot's wrist. This camera is used for visual servoing, i.e. to guide the arm towards the object by providing real-time information on the object's location. The camera is able to deliver 200 frames per second (fps) at a resolution of 640x480.

5 Control Software

5.1 Manipulator Control

Figure 4 shows the main processes running on both boards. Visual Servoing and Object Recognition modules running on the *Vision Computer* provide the *Manipulation Computer* with real-time information for guiding the arm towards an object, or to initiate proper actions according to the context. The Motion Generation module implements the CAN bus communication that interfaces with the arms, the direct and inverse kinematics algorithms, and controls the task execution as well as the communication between the two computers.

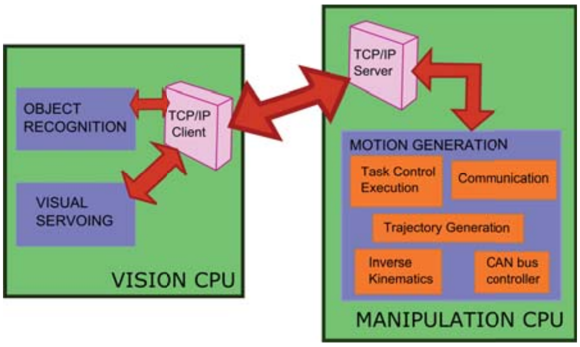


Figure 4. Manipulator software architecture

5.2 Object Recognition

In this section, we describe the techniques used to find objects in the images provided by the cameras as well as their usage within our demonstration scenario.

Matrox Geometric Model Finder For the object detection, we employ the MIL Geometric Model Finder (GMF) included in the Matrox Imaging Library (MIL) 7.0. A model is defined by a set of geometric primitives, extracted from a real picture of the object to recognise. In order to emphasize edges, both weighting of edges and masking of irrelevant areas is also performed on these pictures.

Detecting Light Sources The detection of the light is performed using the left lens of the stereo camera mounted on the robot's head. We capture images from the camera with very low exposition, so that only high-energy light signals are perceived by the camera. After that, a threshold operation is performed, where higher grey values are mapped to white, and lower grey values to black. By counting the white pixels of the thresholded image in a given region of interest of the image, we determine if there is an active light source in the given region of interest on the image.

Usage in the Demonstration Scenario As described in Section 2, the robot system responds to a light signal that indicates an error state of the $SmartFactory^{KL}$. Currently the position of the light signal relative to the robot is fixed. Thus, we apply the light detecting algorithm to a predefined region of interest within the camera image.

A second task is to find an empty carrier travelling on the conveyor belt. In this case, a fixed known area exists that the empty carrier has to cross. The left lens is focussed on this position and the object detection operates with a trained model of features of an empty carrier in order to detect it.

To identify the returning carrier holding an emergency pill container, the object detection has to distinguish our 'emergeny container' from all other objects (normal pill containers, soap containers, etc...) on the production line. For that purpose, the 'emergency pill containers' are equipped with a distinctive label, clearly seen on Figure 5 (b). The scanned area on the production line is the same as the one for looking for the empty carrier. Though this time a model of a pill container is defined by emphasizing features of this special label and masking irrelevant regions.

5.3 Visual Servoing

A design requirement is that our robot is able to grasp objects from non-predetermined positions. Therefore, we implemented a visual servoing strategy that guides the manipulator towards the object to handle. This section first describes the visual servoing task within the demonstration scenario as well as its technical realization.

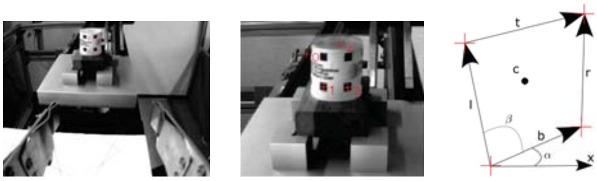
Scenario Within the demonstration scenario there are three situations when the manipulator interacts with an object: 1) a carrier is stopped by the robot's right arm, 2) an empty 'emergency pill container' is grasped with the left arm and placed on the empty carrier, and 3) the filled 'emergency pill container' is removed from the production line using the left arm.

Visual servoing is used in steps 2 and 3. The images employed are provided by a high-speed Prosilica camera which is mounted onto the robot's arm, as described in the previous section.

We assume the object to be visible by the wrist camera during the visual servoing operation. Moreover, we assume that no adjustments on the gripper's orientation are necessary for a successful grasp.

Technical Realization To perform visual servoing, we extract features from the object's geometric properties. These features are then tracked in subsequent frames. The intrisic parameters of the camera are known, so we can calculate the distance from the gripper to the pill container for all Cartesian axes of the manipulator's coordinate system. These distances are then minimized by moving the manipulator towards the pill container.

There are two major concerns that we dealt with: first, to deal with background 'noise' (e.g. people moving in the background) that might disturb the tracking of the image features, and second, to provide robustness against changing light conditions. For those reasons, we chose features that are enclosed within the pill container, with known geometric properties, and that are reliable to track. We used labels for the pill containers that provide dark squares with a fixed and known size on a bright background. The pill container is expected to be always found in such a way that the center of four dark squares points towards the front with respect to the end-position of the visual servoing. Figure 5(a) shows a view from the wrist camera and the (correctly initialized) tracking points. Figure 5(b) shows a possible initialization error of the points to track. For that reason, the plausibility check shown in Fig. 5(c) was included.



(a) (b) (c)

Figure 5. (a) Correctly initialized tracking points. (b) Wrong tracking points (c) Plausibility check

Plausibility Check To verify that the tracking points are both correctly initialized and not lost, a plausibility check was implemented. It is based on the known geometry of the dark squares painted on the label of the pill container, as depicted in Figure 5(c). If the camera is perfectly in front of the pill container, $\mathbf{t}$, $\mathbf{r}$, $\mathbf{b}$ and $\mathbf{l}$ construct a rectangle parallel to the image plane.

Tracking To track the image features in successive frames we use a Kanade-Lucas-Tomasi (KLT) tracker from *ViSP*[3], a wrapper for the KLT feature tracker implemented in OpenCV. We initialize the features to track by locating the pill container using GMF. After the dark squares on the label are located, the KLT tracker features are initialized with the centers of each square.

5.4 Communication and Task Control Execution

As previously described, one of the computer boards is dedicated to process vision sensor data, while the other board directly controls the actuators, i.e. the two manipulators together with the pan-tilt unit. The manipulator coordination is the core task but requires permanent access to information from the vision sensors. Since these two processes run on different machines, a high-level communication protocol is required that allows an efficient bidirectional communication. This communication uses a high-level TCP/IP based protocol. For the given scenario, a request-response communication protocol has been defined which allows communication between modules with low overhead.

Services and their functionalities are globally known in advance. Thus, a request simply requires a service identifier and a command identifier; no payload data is considered for requests. To guarantee task coordination between the two computer boards, and in order to provide information about the successful accomplishment of an operation, a command message always expects a response. In contrast to requests, responses will also transport payload data in order to publish communication endpoints and to access continuously updated data required for visual servoing.

Information integration takes place on the *Manipulation Computer*, as well as the control of the actuation process and task execution. The robot uses two main elements for task execution: (a) a task library and (b) a state machine. The task library allows the usage of simple or complex high-level tasks, where each task represents a predefined sequence of robot actions. Simple tasks command the manipulator(s) directly. Complex tasks also require sensory input from other processes, and thus require communication between the processing units for coordination of the control task. The state machine structures the control flow and organizes the sequence of tasks.

6 Experimental Phase

As previously described, the robot steps through four main states: (a) light detection, (b) empty carrier recognition, (c) visual servoing, (d) recognition of emergency pill container coming back. In most states, the robot moves towards known positions. Exception are the two situations where visual servoing is used: in order to grasp the pill container from the table, and when the pill container is removed from the carrier. In both cases the robot is fed with real-time information from the wrist camera about object's location.

Figure 6 shows a series of snapshots during an nonstop fault-recovery demonstration with one of the real modules of the $SmartFactory^{KL}$ in Kaiserslautern (Germany).

Snapshot (A) on Figure 6 shows the robot's head looking towards the lights on the $SmartFactory^{KL}$ module. At snapshot (B), the orange light is detected. The head moves towards the scanning area where the empty carrier should be detected (snapshot (C)). At snapshot (D),

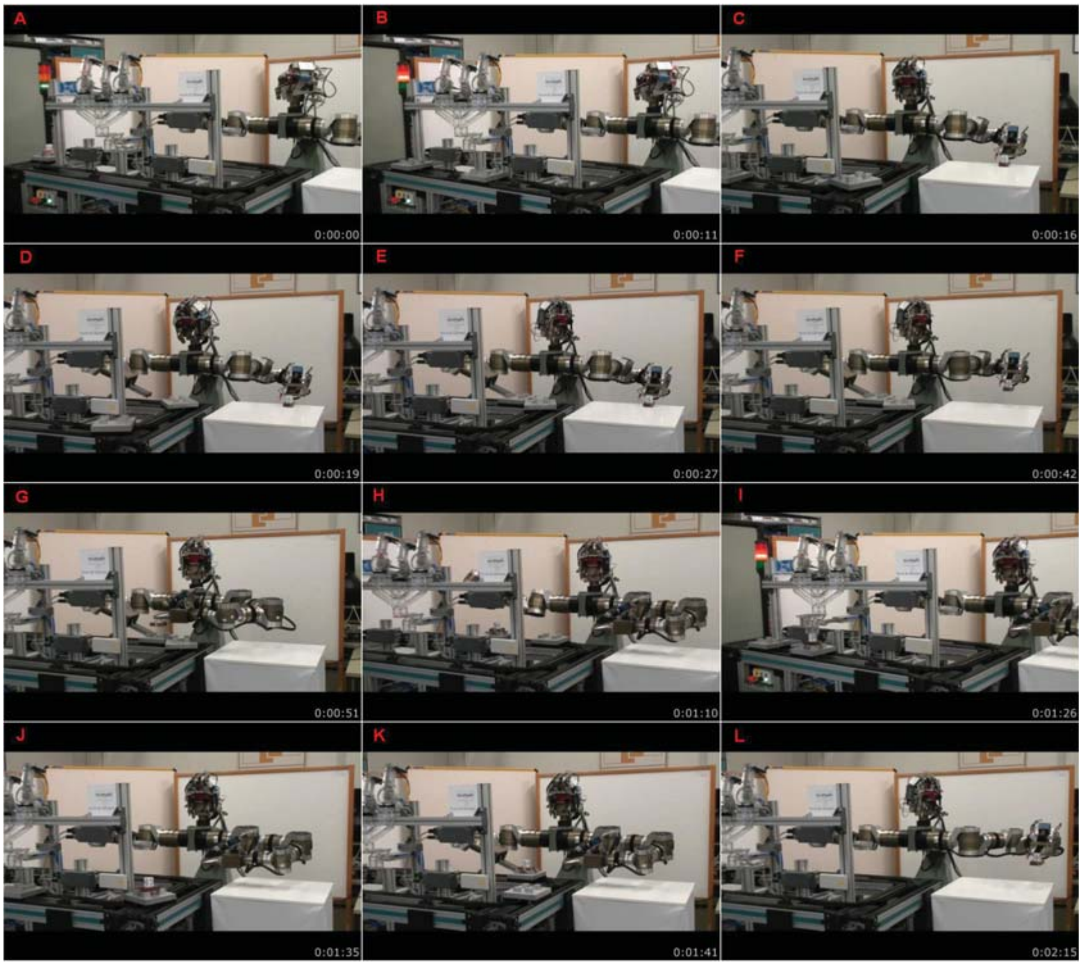


Figure 6. Manipulation demonstration with the real $SmartFactory^{KL}$ module

the right arm goes down to stop the carrier. Snapshots (E)(F) show the visual servoing phase, where the left arm grasps the emergency pill container on the table. Snapshot (G) shows the insertion of the pill container onto the carrier. After that, the right arm releases the carrier (snapshots (H)(I)). The robot's head will move again to wait for the emergency pill container coming back. Snapshot (J) shows the moment when the pill container is detected. The right arm will go down again to hold the carrier (snapshot K)). At snapshot (L), the left arm has removed the pill container from the line and is bringing it to the table, whereas the right arm has already released the carrier, finishing the fault-recovery use case.

7 Outlook

Future work will focus on enhancing the system with a mobile platform. For that reason, a holonomic mobile platform is being designed where the dual-arm system will be mounted on. This will provide the robot with the capability to autonomously navigate and supervise the state of, for instance, the $SmartFactory^{KL}$, locate/perceive problems, drive towards them, and perform the necessary handling operations to solve the contingency.

Acknowledgment

The work presented in this paper is part of the Semantic Product Memory (SemProM) project, funded by the German Federal Ministry of Education and Research (BMBF), grant No. 01IA08002.

References

[1] Motoman SDA10 Dual-Arm Robot. *http://www.motoman.com/datasheets/SDA10.pdf*. As in June 2009.

[2] Trends and challenges in industrial robot automation. *EURON White Paper, Fraunhofer IPA*, (DR-13.4), 2007.

[3] F. C. E. Marchand, F. Spindler. ViSP for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics and Automation Magazine, Special Issue on 'Software Packages for Vision-Based Control of Motion'*, pages 40–52, December 2005.

[4] D. Zuehlke. SmartFactory: from vision to reality in factory technologies. *Proceedings of the 17th IFAC World Congress*, Seoul, Korea, July 2008.