

Design Specification

Joel Fåk, Viktor Eriksson, Per Boström, Eric Gratorp,
Astrid de Laval, Sven Ahlberg

Version 2.0

Status

Reviewed		
Approved		

PROJECT IDENTITY

CDIO, Group 2, Identification of LCD monitors
Linköping University

Name	Responsibility	Telephone	E-mail
Joel Fåk	Project manager	073-425 86 68	joeka281@student.liu.se
Viktor Eriksson	Document manager	070-468 07 15	viker942@student.liu.se
Eric Gratorp	Test manager	070-899 25 42	erigr222@student.liu.se
Sven Ahlberg	Design manager	076-127 06 78	sveah621@student.liu.se
Per Boström	Quality manager	073-248 58 59	perbo612@student.liu.se
Astrid de Laval	Scrum master	070-119 21 69	astde117@student.liu.se

Group mail: cdioproject3@googlegroups.com

Customer: IEI, Linköping University, 581 00 LINKÖPING,
Contact person: Kristofer Elo, kristofer.elo@liu.se

Table of Contents

<u>Document History</u>
<u>System Overview</u>
<u>Kinect Interface</u>
<u>Calibration</u>
<u>Image Processing</u>
<u>Segmentation</u>
<u>Identification</u>
<u>Monitor Selection</u>
<u>Planning</u>
<u>Robot Communication</u>
<u>Main Controller</u>
<u>Graphical User Interface</u>
<u>Robot Movement</u>
<u>Robot Programming</u>
<u>Master Program</u>
<u>Calibration Program</u>
<u>Pick Up Program</u>
<u>References</u>

Document History

Version	Date	Performed changes	Performed by	Reviewed
v0.1		First draft	EG, PB, SA, JF, AdL, VE	

Introduction

2 million LCD monitors & TVs are sold each year in Sweden. As a monitor contains both environmentally harmful substances and valuable materials, the interest in an automated recycling process is large. This project is a part of the research project HÅPLA which investigates the possibility for such processes and is performed as a part of the course TSBB11¹ Images and Graphics at Linköping University. The objective of the project is to detect and locate LCD monitors in a pallet, pick them up and place them by the side of the box. This document will provide a brief overview of how the system will be constructed.

System Overview

To be able to detect and locate monitors, the system will include a Microsoft Kinect sensor that provides 2D-images and depth images. These images will be processed in a computer in order to decide how an industrial robot shall be controlled to pick up a monitor. The diagram in figure 1 is an overview of how the parts are connected.

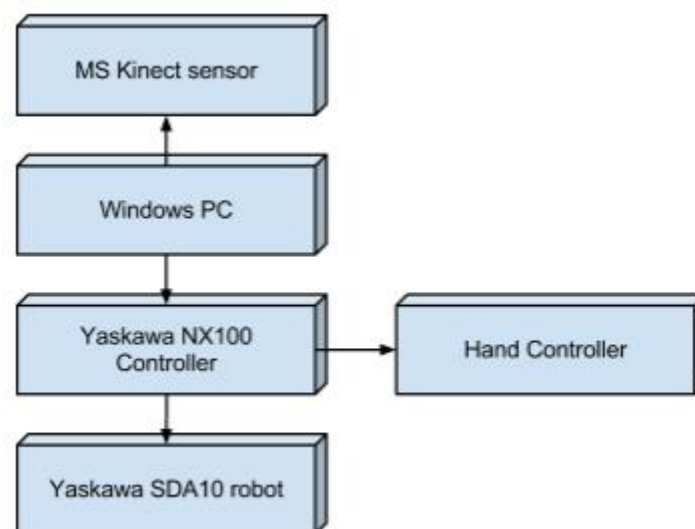


Figure 1: Deployment diagram

The industrial robot that the system will use is a Yaskawa SDA robot which is controlled by the control system Yaskawa NX100.

Kinect Interface

Image data from the Kinect sensor needs to be imported to the computer. To do this, software from

OpenNI² will be used. The RGB image and the three dimensional point cloud are saved to files every time a new calibration or segmentation is started.

Calibration

To be able to move the robot arm to a certain point in space corresponding to an arbitrary point in the Kinect image, a mapping between the image coordinate system and the robot coordinate system needs to be performed. This is done by moving the robot arm between some preselected points in the robot's world coordinate system. The Kinect will be able to find the corresponding points in the camera coordinate system. Using these point-pairs the calibration can be performed using an estimation of the rigid transformation corresponding to the rotation between the two coordinate systems.

In addition to this, it is also a small displacement between the Kinect depth image and the Kinect 2D-image. A calibration between the two images' corresponding points will be performed in order to calculate the intrinsic camera parameters.

Image Processing

The Kinect sensor provides the system with both an RGB image, a depth image and a point cloud of the pallet containing the monitors. These images are used as input to the image processing unit. The main purpose of the image processing unit is to locate the monitors in the pallet and decide which monitor that is most suitable to pick up. A simple flowchart of the input and output of the image processing unit is displayed in figure 2.

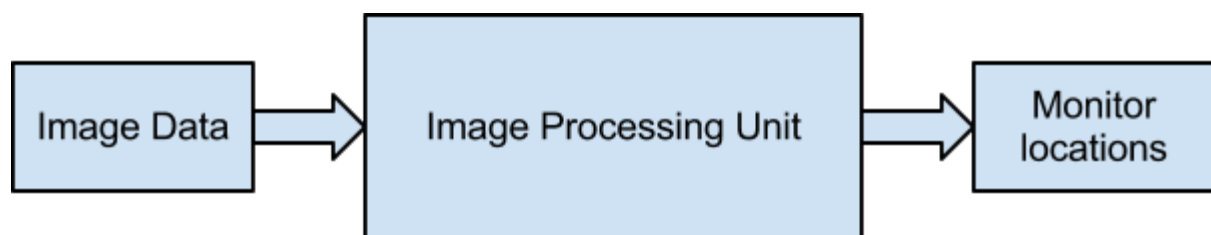


Figure 2: Flowchart of the image processing unit

Segmentation

The first step after receiving the raw image data is to find segments in the 2D image that are likely to be parts of monitors. In order to prevent the floor and the bottom of the pallet from being

recognized as planar surfaces, all these points are removed from the point cloud before further processing. When this is done, the system calculates the normals of each point in the point cloud by performing PCL's normal estimation using integral images with the covariance method. When the normal in each point in the point cloud has been estimated, planar regions are found by clustering these normal vectors. This method returns all the planar regions in the scene, not only the monitors. Therefore, planar regions corresponding to the pallet have to be identified and removed from the monitor planes. The sides of the pallet are modeled as planes that are almost vertical, leading to a z-component that is almost zero in the plane's normal. This makes it easy to pick out all the vertical planes in the scene.

Identification

As the image data have been segmented, the system has to decide which of the detected planes are monitor screens and which are not. Criteria to take in consideration and that every plane corresponding to a monitor screen should meet is for example that the plane should be large enough to be or be a part of a monitor screen and small enough to not be a pallet wall. Planes that do not meet these criteria will be ignored and planes that do will so far be seen as monitor screens.

Monitor Selection

The system will be built to pick up one monitor at a time. Therefore, a comparison between the identified monitor screens is necessary to determine which monitor is the most suitable to pick up. There are several variables that affect the decision.

- A monitor that lies high up should be easier to pick up than one further down in the pallet.
- A large monitor should be better to pick up than a small one since it frees more space in the pallet.
- A monitor that is rectangular, i.e. looks like a standard monitor is less likely to be covered by another monitor than one that is not rectangular. Therefore, a rectangular monitor should be more suitable to pick up.

To find the most suitable monitor to pick up, it is necessary to first decide the visibility and validity of the monitors and finally compare the monitors with respect to how high up in the pallet they are located.

In order to determine the visibility of each located monitor, a box drawn around the point cloud and the convex hull of the point cloud is compared. Using a predetermined threshold and these two areas, it can be determined which monitors that are visible and which are not.

After the previous visibility test the next step is to it is important to check whether the visible monitors that are really monitors and not just a small part of a monitor or a pallet wall. To do this, the following tests have to be passed:

- The area of the monitor's convex hull must exceed 700 square centimeters.
- The shortest side of the monitor's bounding box must exceed a certain value. In this case 20 cm was chosen since the diameter of the grasping tool is about 19 cm.
- The longest side of the monitor's bounding box must be below 60 cm. Otherwise the monitor is most certainly a side of the pallet.

When the unsuitable monitors have been filtered out and the validity of the monitors have been checked, the remaining monitors' value in the z-direction (upwards) are then compared to decide which monitor located on top in the pallet.

Planning

From the image processing unit, the monitor most suitable to pick up is given. To pick up a monitor, the robot needs instructions of where and in what direction to grasp. A simple flowchart of the input and output of the planning unit is displayed in figure 4.

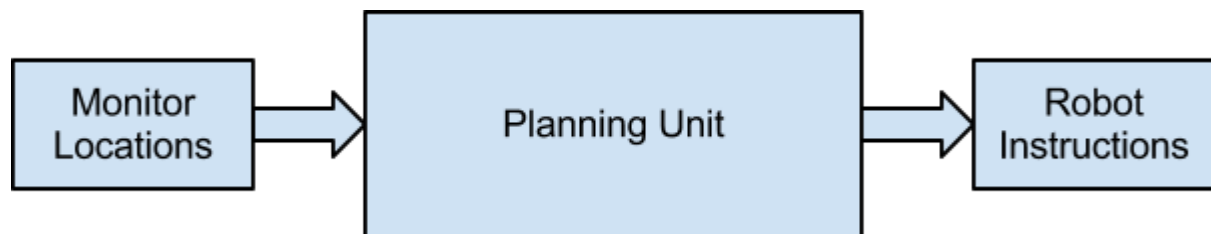


Figure 3: Flowchart of the planning unit

When it is determined which monitor is most suitable to pick up, a route from the robot arm's position to the grasping point of the monitor needs to be calculated. This route consists of three coordinates:

1. One out of three pre selected starting positions
2. Grasping tool aligned with monitor screen's normal
3. Tool placed a few centimeters above the monitors grasping point

The grasping point is simply selected as the center of the monitor screen's bounding box.

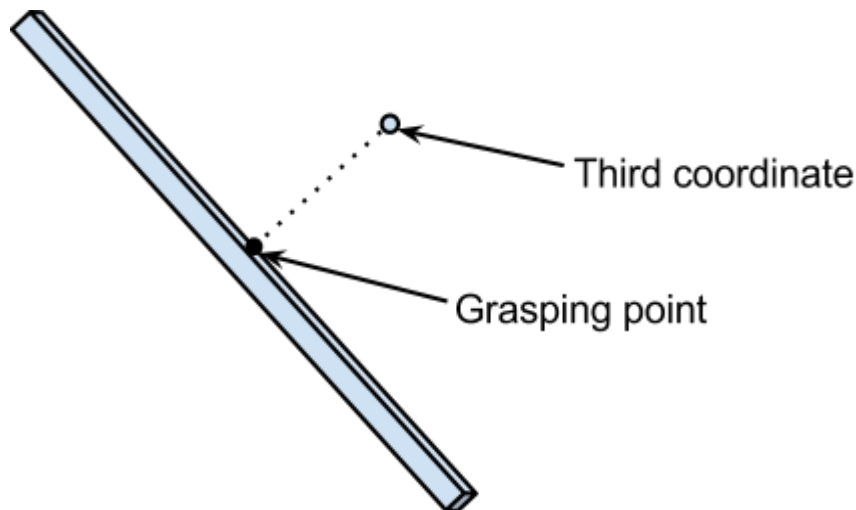


Figure 4: Illustration of the arm's last movement before pick up

Robot Communication

The Yaskawa Controller runs a web server and the communication is performed by http requests. The robot communication unit is implemented as a class with several methods. It connects to the Yaskawa Controller and establishes a session when the program is started. This session is maintained during the whole execution of the program.

The other methods creates the correct http requests for asking the Yaskawa Controller of the current status of the robot or for setting variables. Since the robot coordinate system differs from the Kinect coordinate system, all positions and angles has to be translated. This is done in the calibration unit by request of the robot communication unit.

Main Controller

The main program connects all the different units and handles all transmission and reception between the robot and the program. Multiple threads are used to speed up the system. By doing so it is possible to start the image processing as soon as the robot arm is out of the Kinect's field of view. A flowchart of the main program is displayed in the figure below.

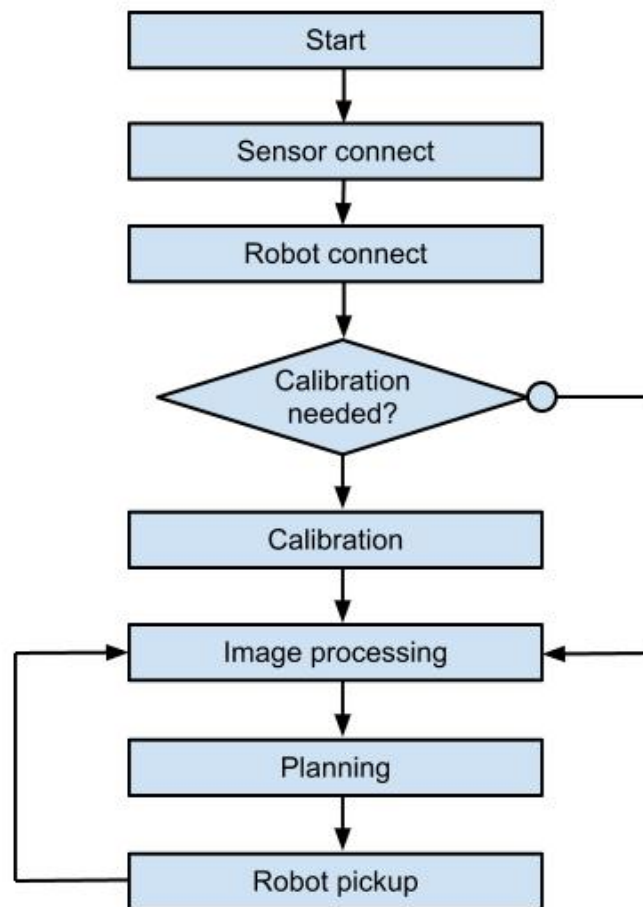


Figure 5: Communication between units

Graphical User Interface

The Graphical User Interface, GUI, will be able to start and stop the system and chose whether to calibrate or pick up. If the system encounters an error, information about it will be displayed here.

Robot Movement

The robot unit consists of Yaskawa NX100 Controller, a hand control and Yaskawa SDA10 robot. The robot communication unit provides the robot with instructions. These instructions contain the set of coordinates that the robot arm, or more specifically the tool center point, should move through to get to the grasping point of a desired LCD monitor. The robot arm consists of seven axis. Three cartesian coordinates and three rotations define a 3D point together with orientation.

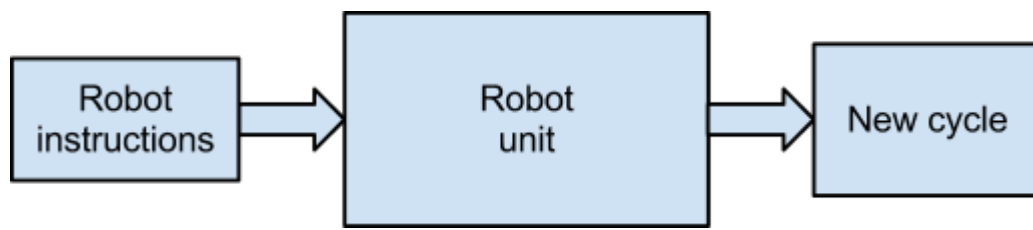


Figure 6: Flowchart over robot movement process.

Robot Programming

The hand control is used to create a program which tells the robot to execute a certain instruction at a certain point in time. This program will control the robot's movements and actions. Below follows the structure of the robot program.

Master Program

Moves the robot to a starting position, and then chooses a sub program depending on input from Robot communication unit.

Calibration Program

Moves the robot arm through a series of predetermined points. Upon reaching each point, the program waits for the kinect to take a snapshot and for that snapshot to be processed.

Pick Up Program

Moves the robot through a set of predetermined and arbitrary points. The predetermined points include a starting point, a drop off point and a set of points which will force the arm to move through a certain path to avoid collision with the pallet and the kinect. The arbitrary points specify the robot's movements inside the pallet. How and why these points are calculated can be read in the *planning section*.

When the image processing is done and which monitor to pick up has been determined, the Image processing unit notifies the Robot unit. The robot arm is positioned above the pallet. The path from this point down to the grasping point of the monitor is decided by the arbitrary points received from the Image processing unit. When the grasping tool has reached the grasping point, the robot arm will move backward through the arbitrary points, in order to place the monitor at a user chosen drop off point. Below follows a flow chart of the pickup cycle.

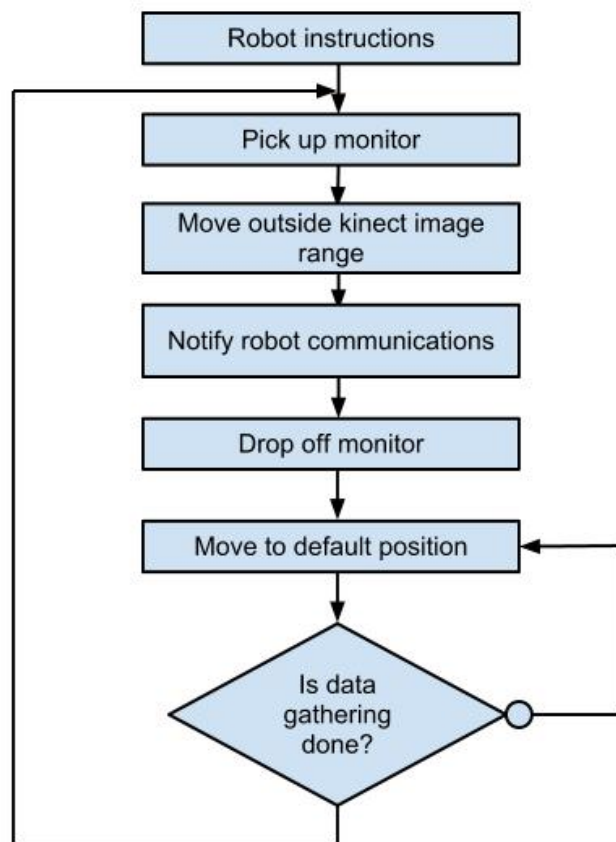


Figure 7: Flowchart over robot unit.

References

The references used in this document are listed below.

- ¹Linköping University, ISY, (2012), *Project Course TSBB11*
<http://www.cvl.isy.liu.se/education/undergraduate/tsbb11>
- ²PrimeSense et al. (2009), *Open Source Natural Interaction*
<http://openni.org/>
- ³Achanta, R, et al. (2012), *SLIC Superpixel Compared to State-of-the-Art Superpixel Methods*
http://ivrg.epfl.ch/supplementary_material/RK_SLICSuperpixels/index.html
- ⁴Achtert, E, et al. (2006), *Finding Hierarchies of Subspace Clusters*
Institute for Informatics, Ludwig-Maximilians-Universität München, Germany

