AUTOMATED SERVER REPAIR FOR DATA CENTERS USING TWO 7-AXIS ARMS AND MACHINE VISION

STUDENTS: MARCUS CHU, WICHWONG PREMVUTI, IAN GOOD, KHAI PHAM

Microsoft

09

Software Architecture

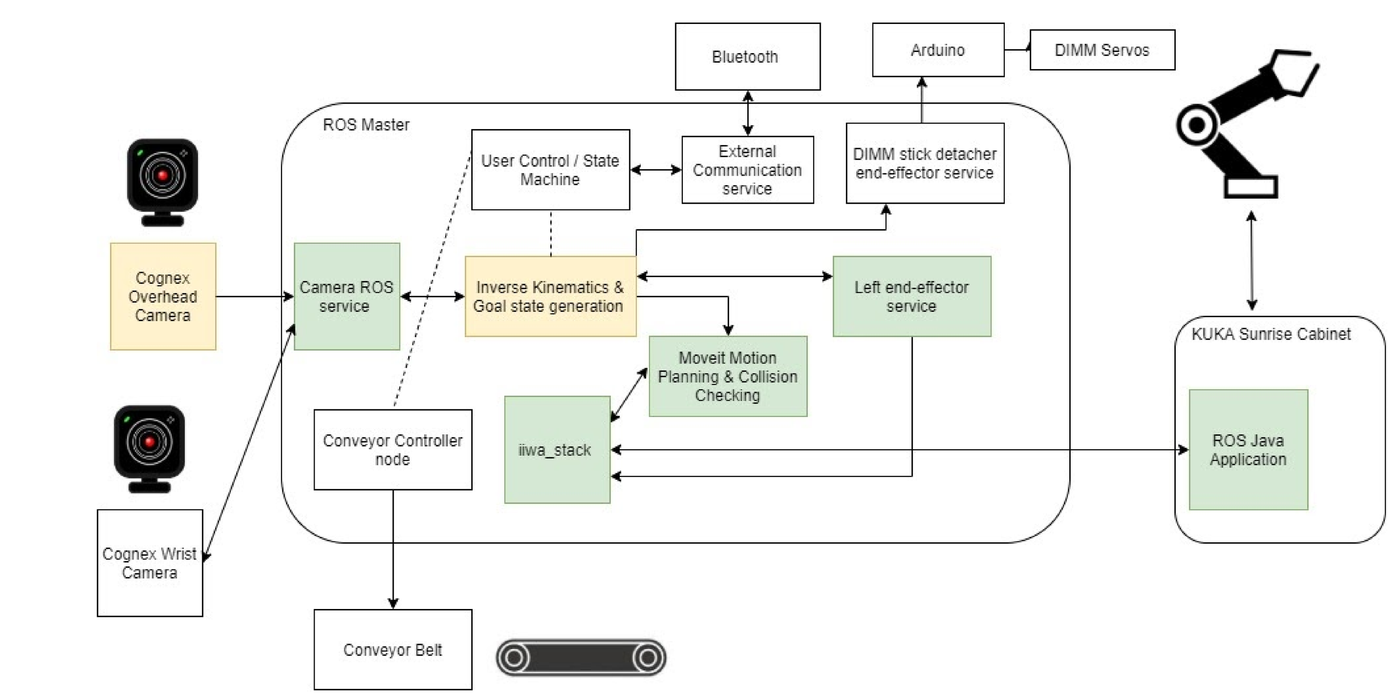


Fig. 1: Overall block diagram for system software architecture.

- ROS was chosen as the framework for inter-process communication and package management.
- Separating processes into nodes allows for granular organization, all controlled from the ROS master. Since our system is distributed across several computers and multiple devices, having an organized structure is a requirement.
- Integration with KUKA controller was a challenge as there was no official ROS support.
- Used ROSJava: a ROS distro ported to run in the JRE on the KUKA controller to interface with the arms.
- Additionally, *iwa_stack*[1] was used as a ROS package that provides integration for KUKA arm execution.

Motion Planning & Simulation

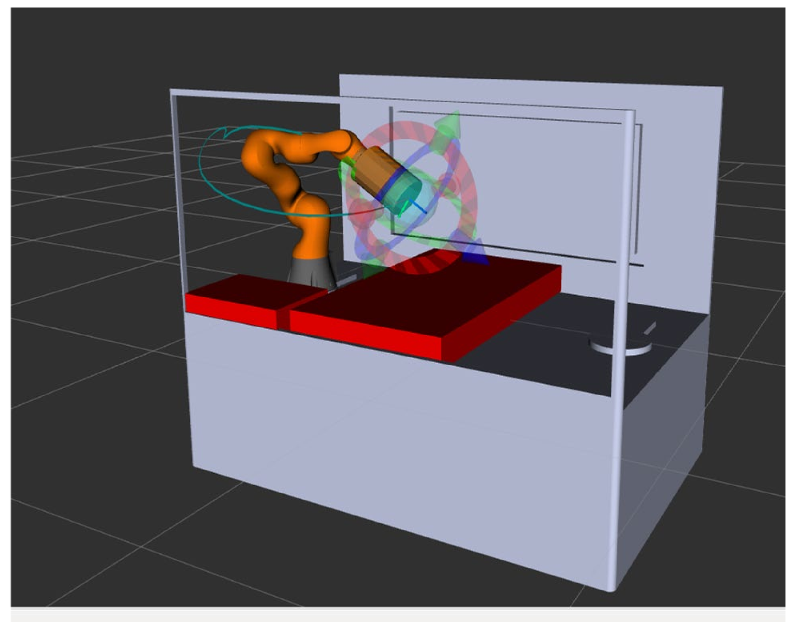


Fig. 2: Simulation scene generated in MoveIt

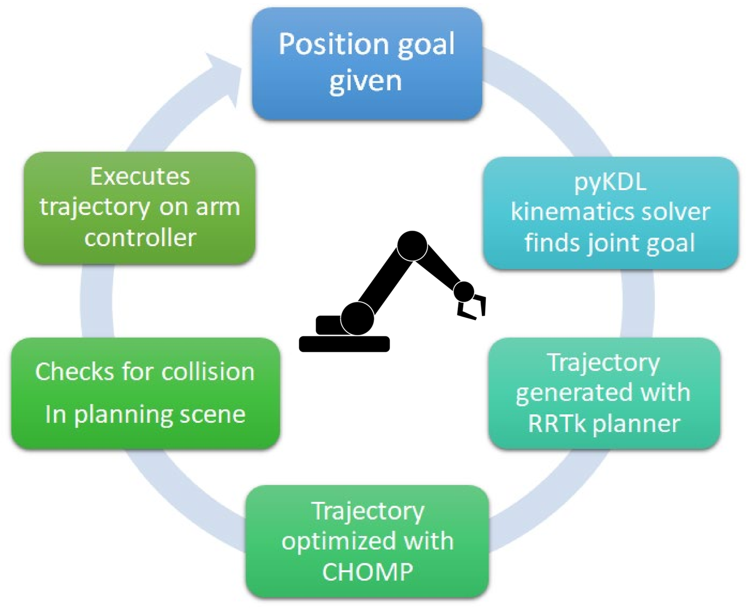


Fig. 3: Process flow for trajectory generation

- To access a goal with variable pose, motion planning was utilized of teaching.
- A simulation scene was modeled and generated using MoveIt!, a ROS package for simulation and motion planning.
- Plans can be modeled and executed directly from the simulation with *iiwa_stack[1]*.
- OMPL provides plugins for most motion planning algorithms (RRT, PRM, etc.).
- Motion planning requests can be post-processed to optimize trajectory.
- Gradient-based planners such as CHOMP[2] used in post-processing provide smoother trajectories than conventional planners alone.

Introduction

Manually repairing servers is **labor intensive** and **costly** for large server farms. By implementing a work cell that can **replace parts on a server** with minimal human intervention, we produced an **autonomous solution** that can help to reduce costs and man hours spent on menial tasks.

With two KUKA 7-axis arms and custom end effectors on a server work cell, we want to prepare for a more realistic use-case scenario by integrating machine vision into the project to handle arbitrary server poses. The work cell will work in tandem with a server pulling autonomous guided vehicle (AGV), where the AGV will pass the server to the workcell onto a conveyor belt.

The Workcell

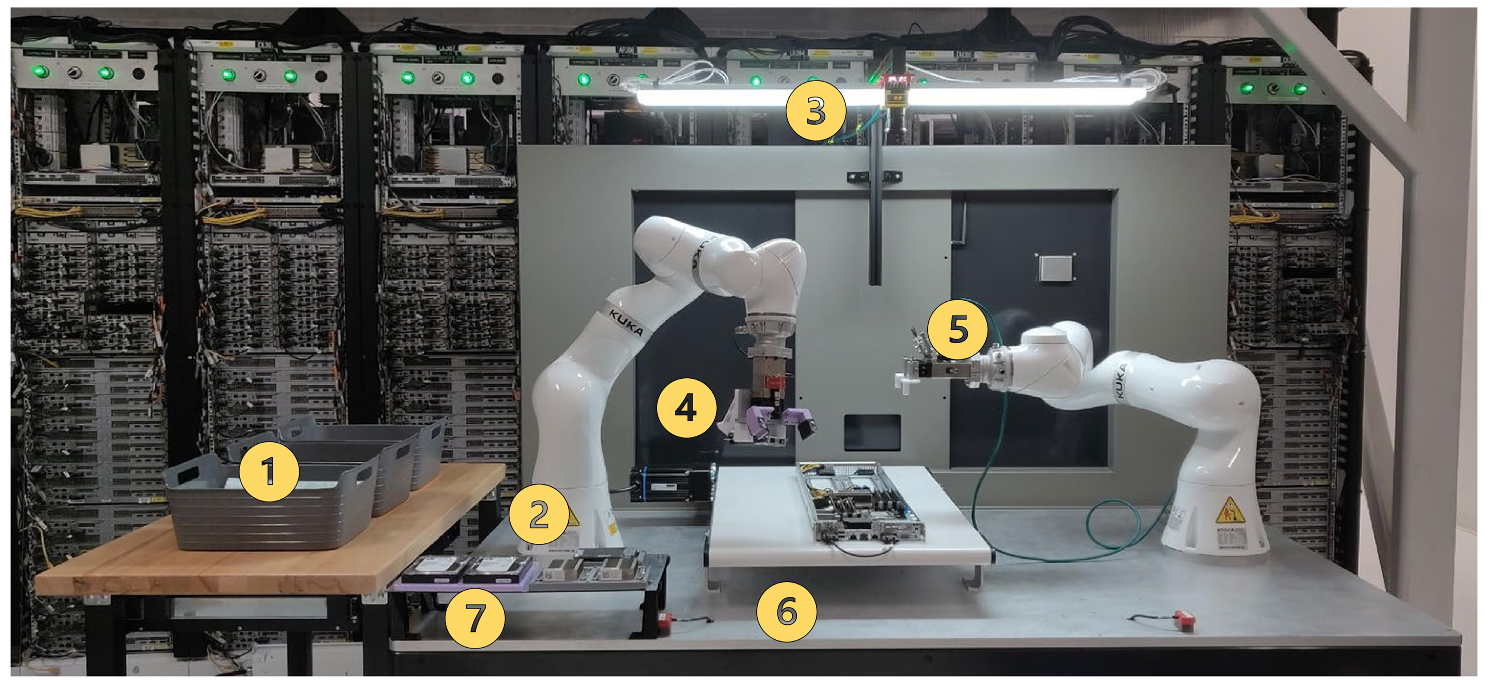


Fig. 4: Image of the workcell with a server positioned

1. Bins for replaced server components
2. Two 7-axis Kuka LBR iiwa 14 arms to replace components
3. Overhead camera for determining server pose
4. Large component end effector
5. RAM End Effector
6. Conveyor belt to bring in server for repair
7. Parts tray for new components

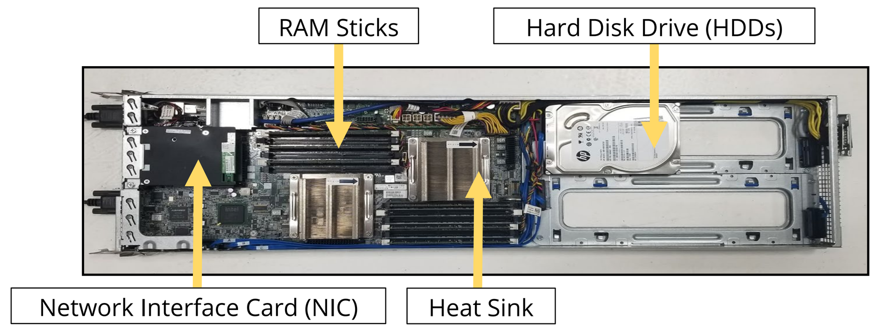


Fig. 5: Diagram of server components to be replaced

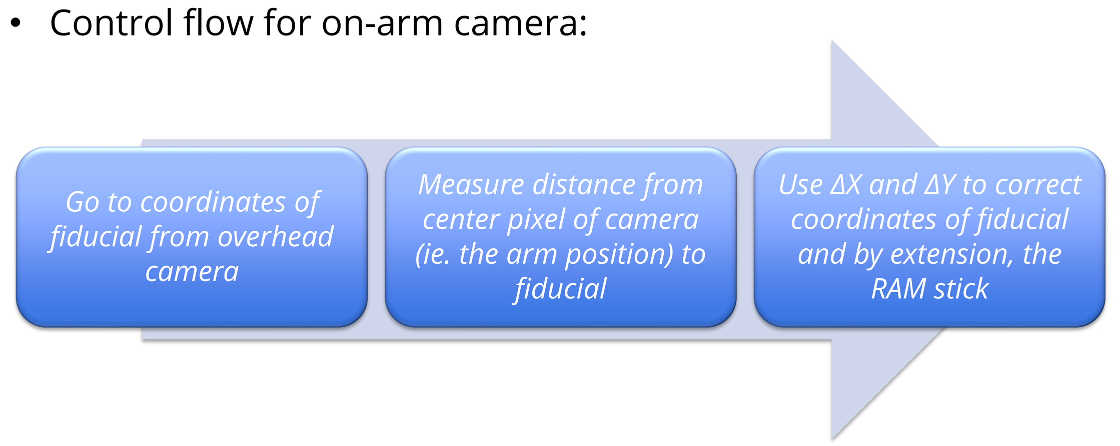
Machine Vision

Overhead camera to provide server pose information

- Using the pose of three fiducials mounted on the server, we can figure out the theta, and XY coordinates of our parts of interest
- Uses Cognex's Pat-Max pattern-matching tool to train the camera to recognize the mounted fiducials

On-camera to provide error-correction for RAM

- RAM sticks require precise and accurate coordinates
- Control flow for on-arm camera:



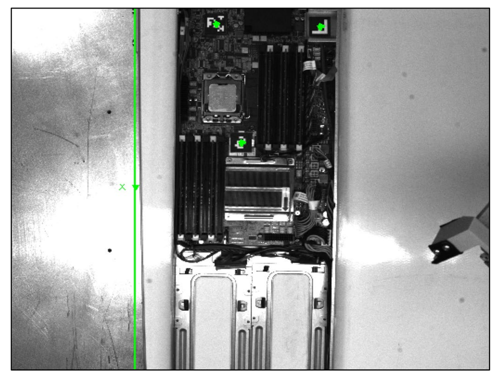


Fig. 6: Image from overhead camera (above)

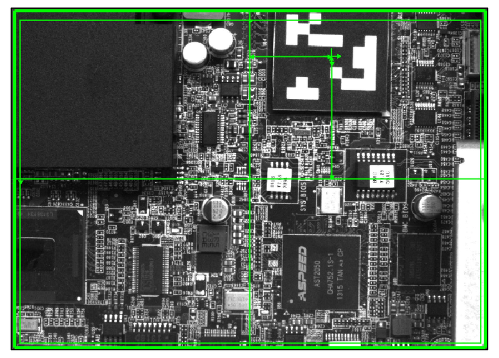


Fig. 7: Image from on-arm camera

Large Component End Effector

- Replaces the NIC, CPU heat sinks, and HDDs
- **Components**
  - Schunk Industrial I/O Actuator
  - Rubber grip pads (black)
  - 3D Printed FDM structure (orange)
- **Pickup Orientation**
  - 30° offset between components was determined via geometric constraints
  - Constraints included collision of the server and the kinematic constraints of the arm

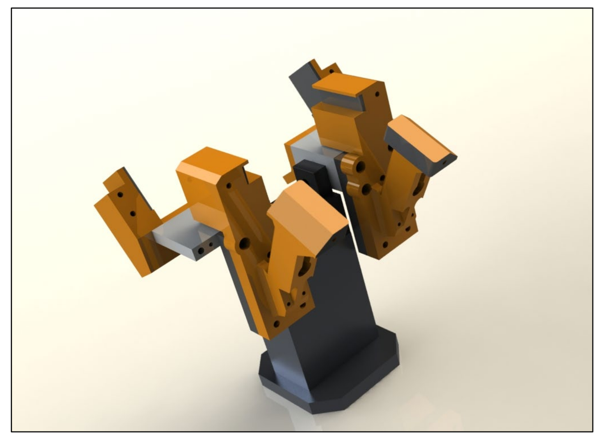


Fig. 8: 3D Render of Large Component End Effector

RAM End Effector

- Replaces the RAM
- **Components**
  - Schunk Industrial I/O Actuator
  - Mechanical Unlock Mechanism
  - Cognex In-Sight 8402 Camera
- **Pickup Orientation**
  - 90° offset gripper to maximize reach of arm, as conveyor belt is offset from center

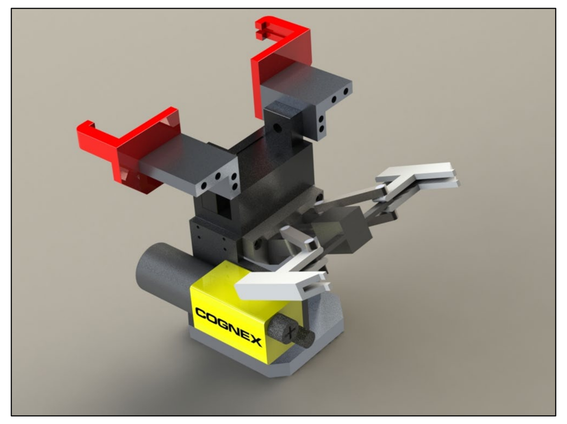


Fig. 9: 3D Render of RAM End Effector

Server Ingress

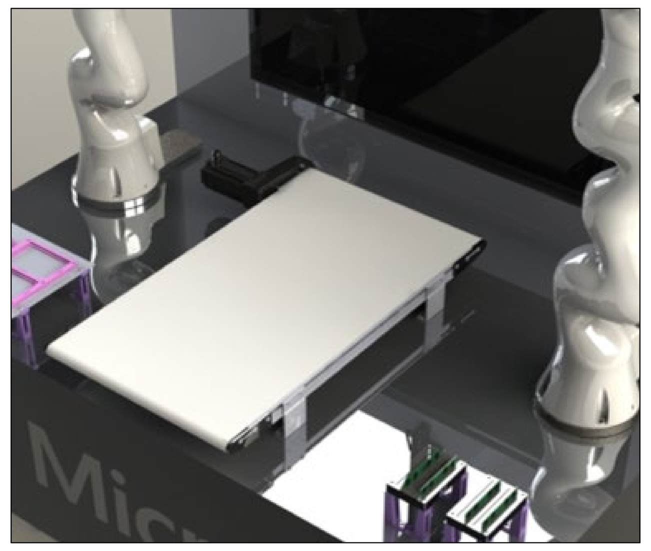


Fig. 10: Render of conveyor belt used to transport server

- Wide Dorner Conveyor Belt to move servers into position
- Driven by ETM MD100 Drive System that is controlled by an Arduino Uno in lieu of a PLC

Communication/Controllers

- Arduino Uno communicates directly to ROS master via Serial
- Conveyor belt automatically stops when the server is in position

References and Acknowledgements

Special thanks to:

- Nicholas Keehn, Corina Arama, and Sean James of Microsoft
- Yana Sosnovskaya, Dr. Howard Chizek, and Rodney Wells of the University of Washington

[1] C. Hennersperger, B. Fuerst, S. Virga, O. Zettinig, B. Frisch, T. Neff, and N. Navab, "Towards MRI-Based Autonomous Robotic US Acquisitions: A First Feasibility Study," *IEEE Transactions on Medical Imaging*, vol. 36, no. 2, pp. 538–548, 2017.

[2] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," *2009 IEEE International Conference on Robotics and Automation*, 2009.

Microsoft KUKA MoveIt! COGNEX

ELECTRICAL & COMPUTER ENGINEERING
UNIVERSITY *of* WASHINGTON

ADVISORS: HOWARD CHIZECK, NICHOLAS KEEHN

SPONSORS: MICROSOFT