Robone: Next Generation Orthopedic Surgical Device

(Check Point Presentation)

Team 1:
Andrew Hundt
Alex Strickland
Shahriar Sefati

Peter Kazanzides (Mentor)
Russell Taylor (Mentor)
Overview

- Project summary
- Software Design and Tool Selection Update
- Completed Progress
- Future Steps
- Dependencies
- Deliverables
- Key Dates
Project Summary

The existing orthopedic surgical device isn’t perfect and requires the patient to be fixed to the operating table, an invasive and time consuming process.

A next generation system will make real time position adjustments using a device such as an optical tracker so fixation is no longer necessary.

The goal is to enable faster and less invasive surgery with this type of computer assisted tool.
Original Software Design

C++ Motion Control Software Stack

Controller PC

Cut File

Cut File Parser

Conversion to series of goal points

Estimate and Apply Offset to Bone Position

Optical Tracker

Convert to Format for Motion Planning Execution

Orcos Kinematics Dynamics Library (KDL)

KDL-FRI Interface

LCSR Controllers Library

KDL v-rep Interface

Reflexes Motion Planning Library

Optical Tracker Update Loop

Ethernet

Java Sunrise Connectivity Suite

Physical KUKA Controller

FRI UDP Software

FRI Java Library Initialization

Motion Safety Barriers

v-rep Sunrise Integration

v-rep C++ API
Current Software Design

Java Sunrise Connectivity Suite

ZMQ_SmartServoCommand.java
FRI Java Library Initialization
SmartServo.setDest()
Google Flatbuffer Serialization
grl::flatbuffer::JointState
JeroMQ Socket

KONI Real Time Interface (Ethernet Cable #1)
ZeroMQ Message via Sunrise Java Interface (Ethernet Cable #2)

C++ Motion Control Software Stack

Controller PC

iwa Class (Driver)
Boost::asio
boost::asio::udp::socket
Kuka nanopb serialization:
FRICommandMessage
FRIMonitorMessage
KukaFRITHreadSeparator Class

driver thread

ZeroMQ
azmq::socket

Google Flatbuffer Serialization
grl::flatbuffer::JointState

AzmqFlatbuffer Class

KukaVrepPlugin Class

vrep thread

v-rep
virtual robot experimentation platform

Optical Tracker

Estimate and Apply Offset to Bone Position

Reflexxes
Motion Planning Library
or
Future Alternative
Motion Planning Library

Plugin
# Software Tool Selection

<table>
<thead>
<tr>
<th>Package</th>
<th>CISST</th>
<th>ROS</th>
<th>ZeroMQ Middleware</th>
<th>Google Flatbuffers</th>
<th>V-REP Simulator</th>
<th>Gazebo Simulator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java and C++ (Req)</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>High Performance</td>
<td>✓</td>
<td>✗</td>
<td>✔︎</td>
<td>✔︎</td>
<td></td>
<td>✗</td>
</tr>
<tr>
<td>Global Adoption in SW Dev Market</td>
<td>✗</td>
<td>✗</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Global Adoption in Robotics Market</td>
<td>✗ mostly @ JHU</td>
<td>☑</td>
<td>ZMQ tools in dev by OSRF (ROS)</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cross Platform</td>
<td>✓</td>
<td>✗</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✔︎</td>
</tr>
<tr>
<td>Well Documented</td>
<td>✗</td>
<td>✗</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✗</td>
</tr>
<tr>
<td>Easy to Use</td>
<td>✓</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✗</td>
</tr>
<tr>
<td>Reliable</td>
<td>✓</td>
<td>✗</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✔︎</td>
<td>✗</td>
</tr>
<tr>
<td>KUKA iiwa integration</td>
<td>✗</td>
<td>✗</td>
<td>-</td>
<td>-</td>
<td>✔︎</td>
<td>✗</td>
</tr>
<tr>
<td>Can add new planning algs</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>✔︎</td>
<td>✔︎</td>
</tr>
<tr>
<td>Final Selection</td>
<td>Possible future integration</td>
<td>✗ ✗ ✗</td>
<td>✓ ✓ ✓</td>
<td>✓ ✓ ✓</td>
<td>✓ ✓ ✓</td>
<td>✗ ✗ ✗</td>
</tr>
</tbody>
</table>
Initial Simulation - Complete

Created a milling simulation in V-REP with a real bone model

- Simulates arm with integrated motion planning

- Simulates cutting bone surface (no interior)

- No optical tracker

- Uses Reflexxess type IV planning library
Kuka Arm Integration - Complete

- C++ Driver
  - Reads arm state from Fast Robot Interface (FRI)
  - Sends commands to Java Connectivity Suite on arm’s controller.

- Java Sunrise Connectivity Suite
  - Initializes the arm
  - Implements asynchronous joint motion commands

- Moves along a specified path in cartesian space

Kuka’s FRI interface for sending commands isn’t usable, so this design works around those flaws.
GRL Library

Published, Documented, and Available

- Kuka LBR iiwa drivers
- V-REP integration

git
https://github.com/ahundt/grl

docs
https://ahundt.github.io/grl/
Live Demo!
Future Steps

Cut file integration

- Acquire ascii cut files
- Implement ascii parsing and conversion to format amenable to sending to planner or arm as commands
- Test parsing and motion commands in simulation
- Test parsing and motion commands on physical robot
Future Steps

Optical Tracker Integration

- Acquire optical tracker
- Add simulation with optical tracker
- Estimate an acceptable response time of optical tracker during surgery
- Setup of optical tracker with kuka
- Implement reading of optical tracker data into software, using existing saw components
- Integrate optical tracking data into cut file and arm commanding loops
- Reaction time testing
  - draw a straight line on an object, move object and check response time
  - see “physical simulation of cutting” idea below
  - Characterize response time
  - Improve response time if necessary
Milling Physical Simulation
Create a physical simulation of cutting, as opposed to a computer simulation.

The initial concept is to put an optical tracker fiducial on the end effector and have a clear box to simulate "bone". We can then use the optical tracker to generate a simulated estimate of actual cutting. This avoids the complexity of acquiring materials to cut and dealing with the dust created by milling foam, wood or other test cutting materials.

- Design and Create fiducial mounting attachment
- Implement logging of physical simulation
- Integrate logging with VREP to visualize execution of simulation
- Implement method to evaluate planned vs actual path within error bounds of sensors
- Create evaluation analysis
Dependencies

- **KUKA robot arm (Resolved)**
  - Access to arm
- **Software (Workarounds Implemented)**
  - Some higher quality level integration of arm control software, such as real time torque control, depends on planned software updates by the manufacturer, KUKA.
- **Logistics (Ok so far)**
  - Access to mentors
- **Optical Tracker (Resolved)**
  - Atracsys optical tracking device
- **Milling device (Max Goal Dropped)**
  - Integration of milling device to the arm
Deliverables

Minimum

✓ 1. Receive arm state in real time

➔ 2. Read in cut file specifying shape of implant

3. Drive both simulated and physical KUKA arm along
   ✓ i. simulated cut file path
   ➔ ii. real cut file path

Expected

➔ 1. Receive optical tracker position in real time and adjust cut path accordingly

➔ 2. Characterize performance

Maximum

➔ 1. Milling Physical Simulation

✗ 2. Investigate arm motion planning

✗ 3. Milling Device Integration
# Key Dates

<table>
<thead>
<tr>
<th>Task</th>
<th>Goal Level</th>
<th>February</th>
<th>March</th>
<th>April</th>
<th>May</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Simulation</td>
<td>min</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
<tr>
<td>Initial Arm Integration</td>
<td>min</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
<tr>
<td>Cut file integration</td>
<td>min</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
<tr>
<td>Optical tracker integration</td>
<td>expected</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
<tr>
<td>Milling Physical Simulation</td>
<td>max</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
<tr>
<td>Investigate arm motion planning</td>
<td>max</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
<tr>
<td>Milling integration</td>
<td>max</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
<tr>
<td>System testing and iteration, Poster Session</td>
<td>expected</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ ]</td>
</tr>
</tbody>
</table>

- *Lighter* colors represent the *original* time frame
- *Darker* colors represent the *updated* time frame

- Green - Completed
- Blue - On Track
- Red - Cancelled
Questions?