Object Tracking by Pan Tilt System

Lily Hoang
University of Tennessee - Knoxville, lhoang1@utk.edu

Follow this and additional works at: https://trace.tennessee.edu/utk_chanhonoproj

Part of the Other Electrical and Computer Engineering Commons, Signal Processing Commons, and the VLSI and Circuits, Embedded and Hardware Systems Commons

Recommended Citation
Hoang, Lily, "Object Tracking by Pan Tilt System" (2014). Chancellor's Honors Program Projects. https://trace.tennessee.edu/utk_chanhonoproj/1677

This Dissertation/Thesis is brought to you for free and open access by the Supervised Undergraduate Student Research and Creative Work at TRACE: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Chancellor's Honors Program Projects by an authorized administrator of TRACE: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.
Final Presentation
of
the Object Tracking by Pan Tilt System

Lily Hoang
ECE 400: Senior Design
Fall 2013
December 12th, 2013
AGENDA

• Introduction: Pan Tilt System Design
  • Overall Design Specifications
• Specific Tasks
  • Goal Tasks
  • Achieved Tasks
• Demo & Walkthrough
  • Demo of Working System
  • Walkthrough of Steps
• Future Work & Conclusions
INTRODUCTION: PAN TILT SYSTEM DESIGN

- Objectives:
  - Improve Pan-Tilt System
  - Implement Object Tracking
  - Redesign User Interface

- Specifications:
  - Track a 6ftx6ft Area
  - Implement with an Android Tablet
  - Positions: Up, Down, Left, Right, and (X,Y) Coordinates
AGENDA

• Introduction: Pan Tilt System Design
  • Overall Design Specifications
• Specific Tasks
  • Goal Tasks
  • Achieved Tasks
• Demo & Walkthrough
  • Demo of Working System
  • Walkthrough of Steps
• Future Work & Conclusions
Task 1: Troubleshoot Camera

1. Troubleshoot Camera
   - Get the camera view to work on Android tablet
   - Fix the resolution problem
   - Incorporate a user setting for the camera IP address
   - Fix the camera mounting issue
   - Fix error handling when the camera connection is not successful
   - Fix the camera reconnect problem
**Achieved Tasks**

- Troubleshoot Camera
  - Got camera view to work on Android tablet
  - Fixed resolution problem
  - Incorporated a user setting for the camera IP address
  - Fixed camera mounting issue

Task 1: Troubleshoot Camera

Task 2: Object Tracking

Task 3: Android User Interface
ACHIEVED TASKS

- Got camera view to work on Android tablet
ACHIEVED TASKS

- Fixed resolution problem
  - Moved all hardware and software from the Netbook to a PC
  - This helped with the video feed processing for the camera system

Task 1: Troubleshoot Camera
Task 2: Object Tracking
Task 3: Android User Interface
**Achieved Tasks**

- Incorporated a user setting for the camera IP address

---

**Task 1:** Troubleshoot Camera

**Task 2:** Object Tracking

**Task 3:** Android User Interface
ACHIEVED TASKS

- Fixed camera mounting issue
  - Developed mount design
  - Built mount and implemented it

Old Mounting (just Tape)
Achieved Tasks

- Developed mount design
  - Measured camera & motor end dimensions
  - Used Google SketchUp to create 3D Models
ACHIEVED TASKS

- Built mount at IRIS West and implemented it
  - 6mm thick ABS plastic and a #8-32 threaded rod

Task 1: Troubleshoot Camera

Task 2: Object Tracking

Task 3: Android User Interface
ACHIEVED TASKS

Old Mounting (just Tape)  New Mount

Task 1: Troubleshoot Camera
Task 2: Object Tracking
Task 3: Android User Interface
**Task 2: Object Tracking**

- **Object Tracking**
  - Learn how to capture frames from the MJPEG stream
  - Develop algorithm to detect objects with captured video frame
    - Detect blob of color
    - Stretch goal: detect laser pointer
  - Develop method to move motors to point the camera at the detected location
**Task 3: Android User Interface**

- Android User Interface
  - **Redesign layout to make it more user friendly**
  - Add help screens with instructions on how to use the PanTiltSee system
  - **Fix Eclipse warnings about string usage, unused APIs, and deprecated preference method**
  - Stretch Goal: Redesign layout so that it will automatically adjust to any screen resolution and orientation (landscape or portrait)
ACHIEVED TASKS

- Android User Interface
  - Redesign layout to make it more user friendly
    - Manual Page
    - Automatic Page/Filter Settings Page
    - Tracking Page

Task 1: Troubleshoot Camera
Task 2: Object Tracking
Task 3: Android User Interface
**ACHIEVED TASKS**

- Redesign layout to make it more user friendly - Manual Page
  - Removed “Show Camera View” button since “Connect Camera” is the same
    - Java: removed the java coding for this button
  - Made font on “Disconnect Camera” larger/consistent

---

**Task 1:** Troubleshoot Camera

**Task 2:** Object Tracking

**Task 3:** Android User Interface
**Achived Tasks**

Task 1: Troubleshoot Camera

Task 2: Object Tracking

Task 3: Android User Interface
Achived Tasks

- Redesign layout to make it more user friendly - Automatic Page/Filter Settings Page
  - Created the general layout of the Filter Settings Page
  - Java: worked with Dave to code this page for the system

Task 1: Troubleshoot Camera
Task 2: Object Tracking
Task 3: Android User Interface
Achieved Tasks

- Filter Settings Page

Task 1: Troubleshoot Camera

Task 2: Object Tracking

Task 3: Android User Interface
**Achieved Tasks**

- Filter Settings Page

---

**Task 1:**
Troubleshoot Camera

**Task 2:**
Object Tracking

**Task 3:**
Android User Interface
ACHIEVED TASKS

- Filter Settings Page
- Mainly Coded:
  - Radio Buttons:
    - Red
    - Yellow
    - Blue
    - Green
    - Violet
    - Orange
    - Custom

```
/** Preset Filter Radio Button Listener **/
    @Override
    public void onCheckedChanged(RadioGroup group, int checkedId) {
        checkedId = group.getCheckedRadioButtonId();
        switch(checkedId) {
            case R.id.radioButtonRed:
                HSVCurrent = Arrays.copyOf(HSVRed, HSVRed.length);
                applySelectedRadioButton(HSVCurrent);
                myFiltered.setHSVFilter(HSVCurrent);
                Log.d("OnCheckedChangeListener","Red Radio Button was selected");
                break;
            case R.id.radioButtonYellow:
                HSVCurrent = Arrays.copyOf(HSVYellow, HSVYellow.length);
                applySelectedRadioButton(HSVCurrent);
                myFiltered.setHSVFilter(HSVCurrent);
                Log.d("OnCheckedChangeListener","Yellow Radio Button was selected");
                break;
            case R.id.radioButtonBlue:
                HSVCurrent = Arrays.copyOf(HSVBlue, HSVBlue.length);
                applySelectedRadioButton(HSVCurrent);
                myFiltered.setHSVFilter(HSVCurrent);
                Log.d("OnCheckedChangeListener","Blue Radio Button was selected");
                break;
            case R.id.radioButtonGreen:
                HSVCurrent = Arrays.copyOf(HSVGreen, HSVGreen.length);
                applySelectedRadioButton(HSVCurrent);
                myFiltered.setHSVFilter(HSVCurrent);
                Log.d("OnCheckedChangeListener","Green Radio Button was selected");
                break;
            case R.id.radioButtonViolet:
                HSVCurrent = Arrays.copyOf(HSVViolet, HSVViolet.length);
                applySelectedRadioButton(HSVCurrent);
                myFiltered.setHSVFilter(HSVCurrent);
                Log.d("OnCheckedChangeListener","Violet Radio Button was selected");
                break;
            case R.id.radioButtonOrange:
                HSVCurrent = Arrays.copyOf(HSVOrange, HSVOrange.length);
                applySelectedRadioButton(HSVCurrent);
                myFiltered.setHSVFilter(HSVCurrent);
                Log.d("OnCheckedChangeListener","Orange Radio Button was selected");
                break;
            case R.id.radioButtonCustom:
                HSVCurrent = Arrays.copyOf(HSVCustom, HSVCustom.length);
                applySelectedRadioButton(HSVCurrent);
                myFiltered.setHSVFilter(HSVCurrent);
                Log.d("OnCheckedChangeListener","Custom Radio Button was pressed");
                break;
        }
    }
};
```

Task 1: Troubleshoot Camera
Task 2: Object Tracking
Task 3: Android User Interface
**Achieved Tasks**

**Task 1:** Troubleshoot Camera

**Task 2:** Object Tracking

**Task 3:** Android User Interface
ACHIEVED TASKS

- Redesign layout to make it more user friendly - Tracking Page
  - Created the general layout of the Tracking Page
  - Java: worked with Dave to code this page

Task 1: Troubleshoot Camera
Task 2: Object Tracking
Task 3: Android User Interface
ACHIEVED TASKS

- **Tracking Page**

![Image of Tracking Page]

- **Task 1:** Troubleshoot Camera
- **Task 2:** Object Tracking
- **Task 3:** Android User Interface
Achieved Tasks

- Tracking Page

This page is currently under development. Please note that this space is reserved for the Tracking System.

Task 1: Troubleshoot Camera
Task 2: Object Tracking
Task 3: Android User Interface
AGENDA

• Introduction: Pan Tilt System Design
  • Overall Design Specifications
• Specific Tasks
  • Goal Tasks
  • Achieved Tasks
• Demo & Walkthrough
  • Demo of Working System
  • Walkthrough of Steps
• Future Work & Conclusions
Demo of General Layout
Demo of Working System - Manual
Walkthrough Steps - Computer

- IP Camera Icon
- PanTiltSee Icon
Walkthrough Steps - Tablet
Walkthrough Steps - Tablet
Walkthrough Steps - Tablet
Walkthrough Steps - Tablet
Demo of Working System - Automatic
AGENDA

• Introduction: Pan Tilt System Design
  • Overall Design Specifications
• Specific Tasks
  • Goal Tasks
  • Achieved Tasks
• Demo & Walkthrough
  • Demo of Working System
  • Walkthrough of Steps
• Future Work & Conclusions
**Future Work**

- Improvements on the camera mount and camera
  - Newer and improved camera models
  - More aesthetic camera mount
  - More secure and stable to avoid vibrations
- Change the device to a newer Netbook
  - Return to the Netbook concept for increased mobility
  - Requires a newer and more updated Netbook with more processing power for the camera stream
- Graphical User Interface
  - Create a help screen for the user
  - Edit the Tracking page after implementing object tracking development
- Object tracking
  - Implement developed C++ code by translating it to Java
CONCLUSIONS

- Troubleshoot Camera
  - Get the camera view to work on Android tablet
  - Fix the resolution problem
  - Incorporate a user setting for the camera IP address
  - Fix the camera mounting issue
    - Fix error handling when the camera connection is not successful
    - Fix the camera reconnect problem

- Object Tracking
  - Learn how to capture frames from the MJPEG stream
  - Develop algorithm to detect objects with captured video frame
    - Detect blob of color
    - Stretch goal: detect laser pointer
  - Develop method to move motors to point the camera at the detected location

- Android User Interface
  - Redesign layout to make it more user friendly
  - Add help screens with instructions on how to use the PanTiltSee system
  - Fix Eclipse warnings about string usage, unused APIs, and deprecated preference method
  - Stretch Goal: Redesign layout so that it will automatically adjust to any screen resolution and orientation (landscape or portrait)