Robot Interaction and Display via The Internet

Authors: Walter Sagehorn and Jonathan Butler

Abstract:

This paper presents an implementation of a web-robot interface that facilitates interaction with ROS-based robot and a user on the internet. The system is designed to accomplish two tasks: allowing researchers to monitor and debug robots remotely, and creating new way for non-researchers to interact with the robots. During an event, like Explore UT, visitors could use the website to view not only where the robot is, but also its surroundings via a live stream. This project seeks to convenience researchers and engage visitors. In the future, the project could be expanded to incorporate more than just data streaming. For example, researchers could give path goals or high-level objective goals through the website. There are major changes that need to be made for that to become a reality, for instance, implementing security measures to insure only those authorized to control the robot can control the robot. If these changes are carried out, this project has potential to become a research tool for the lab at UT and labs across the world.

Introduction:

Having the robots displayed on a webpage for all to see has many benefits. It can be used to ‘wow’ the public or it could be used by those in the lab to monitor the progress of the robots as they go about their given tasks.

Background and/or Related Work:

There has been extensive research done in regards to the combination of robotics and the internet, however, since the topic is so broad there are many directions in which to take it. The paper we found, titled, “A Multimodal Interface to Control a Robot Arm via the Web: A Case Study on Remote Programming” from IEEE, reflects closely to the overall end goal of the project. It details the controlling of arm robots via a web interface, which is something we hope that we can do with our robots and we hope can be applied to not just the segbots, but all of the robots in the laboratory. A statement from the paper, “Enabling remote programming of the robot system permits the operator to develop external programs that take control over the whole set of robotic functions,” explains why it would be beneficial to connect the robots to the web. Completing this functionality would not only be beneficial for our robots but for any systems that would use the code.

Technical Approach:

The hardest part in terms of technical difficulty was the dynamic loading of the topics available to stream. Upon accessing <robot IP>:8080 when web-video-server is running, it returns a stripped-down HTML page that lists the available topics, and links to both the stream and a snapshot. However, the format of the page it links to isn't ideal; we would rather have the url of the stream itself (to provide a the "src" of our <img>, which displays the stream) than a page displaying nothing but the stream (which is what it provides). We've decided to just disregard the snapshot altogether as it wouldn't be very fulfilling to the purpose of this project. The data was processed by the main server process and passed to a dynamic template (using the Jinja2 library) that generated the HTML for the page. But first, in order to extract the information needed from the raw HTML, the python library BeautifulSoup was used to transform the robot response into a Python object that can be manipulated and returned. The data returned wasn't just going to be standard "a href" links-- include information that can be used by a JavaScript (more specifically, JQuery) function that manipulates the source of the video feed. By using this method, we can change which topic is being displayed without reloading the page. This not only improves user experience, it

also prevents the backend from serving the same pages again and again, which would potentially cause scalability issues in terms of server stress.

However, the image source is still the robot's IP, so the traffic is still coming directly from the robot. Considering this, the site is more of a wrapper that surrounds the video feed, enables users to more easily switch between streams, and acts as a easy way to retrieve robot IPs. However, the user experience using the site is considerably better than pure web-video-server. A nice user interface allows on-technical users, like visitors during Explore Texas, to utilize the site as well. Obviously, there are issues with streaming directly from the robot. It wouldn't take much to DDoS a laptop that's already nearly DoS'ing itself. Ideally, the website would request one instance of each stream from the robot, and serve multiple instances from the server to clients.

Another difficulty encountered was the task of parsing the HTML received from the robot. Since the goal is to code in a general and reusable way so that the program doesn't break when future changes are made to other robot code. For instance, let's say that the robot gets a new sensor that publishes to a topic the web-video-server deems a video. We want to not only not break, but also be able to integrate that new topic into the viewing options. To do this, the BeautifulSoup parsing algorithm must be robust and non-assuming.

When analyzing the html output given by the robot, we noticed a possible error in the web-video-server HTML generation code: a <ul> tag immediately followed by a tag. The issue occurs twice in the code output by the segbots running normally. After looking into it, it seems to occur when a topic group is evaluated as video type, but the individual topic are unable to be streamed. For example, /nav_kinect/projector/ is listed as a topic group but contains no topics. It may not be fair to call this an error, but it seems strange to include a category of topics without checking if it contains content.

Most visitors don't carry a laptop with them when touring the GDC, so particular attention was given to mobile UI (user interface). The library Flask-Mobility is an add-on for Flask that detects mobile user-agents and delivers alternative page templates to the user. In our case, the CSS was modified to make some elements appear vertically instead of horizontally in order to fit everything on a narrow screen. The fact that nearly all researchers will access the site through a computer and nearly all visitors will access the site through mobile devices could be taken into account to deliver content that is more suitable for each party.

Evaluation:

The site is more or less functional as intended. The only thing that was stopping us from being able to get everything working and doing a demonstration was the complications with the "nixons-head" server. In theory, the website should run and perform its current function: to display video topics from the robots. We weren't able to do much testing without the server available. Some testing was performed on the robots when the site was incomplete and later, when it was complete, with sample IP JSON and robot replies. Everything worked as intended.

Conclusion and Future Work:

In conclusion, the project turned out well. We did not get everything functioning that we had hoped to, however the progress we made was good considering the difficulties encountered. Specifically, the complications with "nixons-head" prevented sufficient testing and demonstration, but we still completed the basic, working framework of the website. If we could get the server online we could tie up the loose-ends in our project and have a functioning website; for now the simple version is a very good start. As for future work, we hope to improve on the layout of the site as well as its functionality. We hope to add the ability to control, send goal positions etc., and fully display every aspect feasible about our robots onto the webpage. Since we were not able to display

other topics besides video topics, displaying topics such as position and velocities will be our top priority. We think all of this is definitely possible to do and hope that it can become useful for our robots as well as other labs who may find some use for displaying and controlling their robots from the web.

References:

Alexander, B., Hsiao, K., Jenkins, C., Suay, B., & Toris, R. (2012). Robot Web Tools. *IEEE*. Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6377438&tag=1

Marin, R., Sanz, P. J., Nebot, P., & Wirz, R. (2005). A Multimodal Interface to Control a Robot Arm via the Web: A Case Study on Remote Programming. *IEEE, 52*(6). Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1546366&tag=1

Welcome to Flask [Flask Documentation]. (n.d.). http://flask.pocoo.org/docs/0.10/