Brief Article

The Author

August 13, 2016

Abstract

This thesis describes the research, development and construction of a low-cost, tele-operational mobile robot-arm. Development of both a hardware system, and software control interface are achieved, with multiple design issues faced in ensuring a low-cost platform. As a tele-operated system, issues of robot autonomy are avoided, although possible through future development.

1 Introduction

The primary issue faced by the current generation of robot arms is cost. Systems capable of supporting humans, and most importantly, humans with disabilities, are inaccessible, with a minimum cost exceeding £10,000. While low cost robot arms do exist, they are not capable of handeling the complex shapes, or higher weights of objects used on a day-to-day basis by humans. Furthermore, accessible robot arms are small desk-based arms, used for supporting table-top work, not collecting objects from 1-2m heights for a user. The price of effective robot arms also limits the abilities of intuitions or individuals to conduct research and/or development on robot arms, as no low cost, open source platform currently exists. This stagnates the growth of the area, as only limited numbers of robot arms are available to Universities, and these tend to take the form of large scale industrial robots, not mobile arm platforms. By developing an open-source robot arm solution, it becomes possible for hardware and software development to be attempted by anyone, and fed back in to the original design, or continued in parallel with alternative focuses to the main research. Due to this focus, the rules of this research are simplicity of design and simplicity of implementation. This ensures that limited high cost components or manufacturing techniques are needed for creation of the platform, with a high emphasis on 3D printing in both prototyping and the final design.

2 Aims and Objectives

The aims of this research are twofold. To design and develop a low cost (under £2000) mobile robot arm platform, and to design and develop a platform independent teleoperation system for use in manipulating objects within the world.

3 Motivations

4 Research

This thesis covers a range of topics, requiring an effective robot arm design, software control system and electrical system to work in parallel while being robust and low cost. The following research explores the possible issues, and required design considerations in developing this system. The robot arm will be controlled manually and largely lack any form of automation or intelligence, due to this focus, research will primarily examine human controlled robotic manipulators rather than autonomous assistive robots.

4.1 Assistive Robot Arm

[1] presents a robot designed to both support and operate within human environments. A key focus on this, reflected in the design, is the location of objects within human environments "usually found on flat surfaces that are orthogonal to gravity, such as floors, tables, and shelves". Through vertical translation of the robot arm across a centre pylon, the paper states that the same methods for grasping an object can be used for objects at different heights. This compares to multi-jointed robots which must change their gripper behaviour due to differences in approach angle and direction. The robot outlined in [1] uses only two wheels and a caster for motion, while vertical transformation is managed by a linear actuator (one degree of freedom). The gripper has five degrees of a freedom and uses two fingers for gripping. These two fingers are static (not 'soft')

4.2 User Interface

Multiple papers describe the issue of effective user interfaces when developing control systems for robot arms. This issue has been described as of even greater importance when facing scenarios which involve people with disabilities [2]. In [3], experiments are conducted in order to discover the effectiveness of a visually based system versus a menu based interface. What is found is that while manual control is preferred and seen as more accurate by users; menu based control is considered easier [3]. [2] attempted a similar experiment comparing joystick control versus direct selection of an object using a touch screen – the robot arm was not directly controlled, but rather a target was selected on screen as in the menu based system used in [3]. Touchscreen was found to be a more effective form of control, with users simply selecting desired objects [2]; reinforcing [3] where it was found that greater autonomy resulted in less required user input. Furthermore, it was found that control systems where the camera was fixed (independent from the arm's movements) were preferred over systems where the camera moved [2]. Overall, fixed cameras with touchscreen controls led to faster and more accurate operation by users. Interestingly, a user was also examined using a simple keypad method, and found this to be preferred over the joystick due to better accuracy and less drift; commenting also that the touch interface was "the most intuitive". This reflects a final question given to users where joystick sensitivity and drift were found to be the greatest issues. [2] What must also be considered, and as mentioned in [3], is the possibility of users lacking the fine motor controls needed for touchscreen control. Due to this possibility, [3] chose

to implement a system of repeated "quartering" where the image was progressively cut up in to smaller pieces in order for the user to select the desired object (by selecting the "quarter" that held the majority of the desired object). [4], while an older paper, discusses numerous possible issues still relevant in tele-operation, such as image transfer speed, interface simplicity and the overall architecture of server-client based system using web technologies. Similar to the previous systems, the system described in [4] takes additional steps to convert a user's desired destination (through point and click) to robot arm movements. Therefore, as in [2], there is no direct real-time control of the robot arm. In an attempt to simplify control of the robot arm, users were restricted to movement on a 2-dimensional plane, this also allowed for the implementation of the previously discussed point and click methodology. This early research into tele-operated robots via the World Wide Web (using HTML) encountered issues with the transfer rate of images from the server (robot arm) and the client. While these issues were faced using the slower internet and transfer speeds of the early 90s, they must still be considered in today's world.

A further issue faced by the 1994 project was compatibility [4]. While the X window system is described as capable of live video streaming, it was sacrificed in order to provide better compatibility with other systems. The resulting system required manual reloading of the client side image. Other issues concerning compatibility are no longer relevant and consider issues of hardware compatibility and operating system limitations. However, these considerations are similar to those now faced when targeting mobile devices; where architectures and operating systems are wildly dissimilar from their desktop counterparts. Additional levels of robustness are discussed in the paper, specifically a user tutorial to support usage of the system, as well as automatic emails sent when key portions of the system fail. Furthermore, limiting the colour range is also discussed.

Overall the papers discuss an array of issues and exist as useful sources for discovering best practices when developing control interfaces for robot arms.

4.3 Discussion

While [2] and [3] both discuss the strengths of automated or "point and shoot" style systems, the proposed project will be deliberately manual to allow greater interaction with the user's environment, a direction supported by [3] which found manual control was preferred even though it was considered more difficult. The effectiveness of this will have to be examined, however due to the target age of users (children) it can be expected that a greater degree of gamification would be preferable to automation. The effectiveness of touchscreen and keypad controls as found in [2] may suggest that a combination of the two into an on-screen gamepad style control interface may be effective – while also highly familiar to the target group. The use of a 2-dimensional plane by [4] may be effective in minimising user workload and avoiding over complication or frustration. Touch sensors could be used in order to automatically guide the robot gripper forward on the Z-axis. Furthermore, [2] found that a static camera (i.e. not following the gripper) was preferred by users. This will also have to be examined as the proposed project intends to attach cameras to the gripper in order to give the user an effective perspective for easier navigation towards an object. One participant in [2] felt that use of a moving camera "felt more real".

[4] discusses issues faced with image transfer speed. While transfer rates have

greatly increased since 1994, higher resolution screens require higher resolution images. Emphasis must be placed on balancing performance and quality as real-time control will be required by the proposed project, which in turn will require greater frame rate. As the proposed system will involve user manipulation of objects, performance must be favoured to ensure safe and correct usage. Furthermore, slow frame rates may lead to user frustration and inaccuracy – this will have to be examined and an optimum balance discovered. As with the proposed project, compatibility issues existed for live streaming of video [4]. Due to this fact, manually reloading images from the server to the client was required. 12 years later this same technique is still the most robust method when facing different platforms (specifically, mobile ARM based systems and full x86/x86-64 systems). In today's world JavaScript can be used to automate refreshing of the latest image, but effective, web-based, cross-platform systems are still not fully supported. While [4] describes lowering 256bit grey scale to only 64bit, similar limiting of colours could be used to lower the colour palette with alternate file formats (such as GIF) if needed. This may improve the compression currently accomplished with JPEG. This may lead to trials using JPEG2000's region-of-interest based compression systems, however this may limit browser compatibility. Minimal interpolation, or other image processing techniques, can be used on the client side due to the requirement for the system to be platform independent, hence all compression must be either widely supported or permanent (such as lowering the gamut).

Five key points were discussed in [2] describing the needed design specifications of a system designed to be used by disabled people when controlling a robotic arm. While the points described are in reference to an on-board robotic arm attached to a wheelchair, the Human-Robot Interaction considerations still remain highly important.

1. Interface should be easy to use without being trivial.
2. Interfaces should have a minimal length processes. (Referring specifically to cognitive impairments)
3. Interfaces should adjust prompting levels.
4. Interfaces should leverage a person's sensory skills to augment their feedback.
5. Interfaces should accommodate multiple access devices.

These considerations, and those previously discussed, are important as they simplify the use of the system, while maximising productivity and control. Notably, support for multiple devices is mentioned here as well as effective leverage of sensory skills. The proposed project approaches the use of sensory skills through the use of head gestures and natural use of human perception to adjust camera outputs from the system. Adjustment of prompting levels could be implemented as tutorials, time based ques (i.e. user inactivity), or partial automation to correct for user error.

4.4 Conclusion

The papers reviewed were chosen as they effectively covered an array of issues that may be faced in the development of a human-robot interaction system as will be needed by this project. What was found is that some design considerations planned for this project (such as hardware location and software on-screen layout) may not be fully accepted by all users and might be found to be difficult or over simplified by

some. These systems may need to be adjusted or have optional replacements in the future. It is believed that the five points described by [¡Tsui2008¿] reflect an effective development target for this project, while important concerns and vulnerabilities discussed in [3] and [4] will have to be monitored to ensure the system effectively meets all needs both mechanically as well as operationally. In conclusion this review found that specific needs related to the use case scenarios of this project will have to be met in order to succeed, while decade old software solutions may be effective in supporting modern day software issues.

5 Development

This thesis covers a range of topics, requiring an effective robot arm design, software control system and electrical system to work in parallel while being robust and low cost. The following research explores the possible issues, and required design considerations in developing this system. The robot arm will be controlled manually and largely lack any form of automation or intelligence, due to this focus, research will primarily examine human controlled robotic manipulators rather than autonomous assistive robots.

5.1 Software

During development of the system, the software used moved through multiple redesigns in both function and interface. In order to avoid issues of cross-platform portability, a web application was developed as the control system – rather than platform-specific native applications. This avoided additional costs that would be accumulated through development licenses either during development or as a future commercial project. Furthermore, this simplified usage of the control system by only requiring the user to connect to CHAP's ad-hoc wireless network in order to use the system, avoiding additional steps such as installation. The fault in this setup is that it places the effectiveness and reliability of the system entirely in the hands of the user's web browser. With browsers showing extremely mixed results in support of modern HTML5, CSS3, JavaScript and other key web technologies; browser support cannot be relied on. For example, access to a user's webcam through WebRTC (such as for face tracking) is a feature that cannot be guaranteed between browsers. That said, the continued development and advancement of web browsers can be relied upon and isn't prone to the same level of change or redesign visible between both different mobile operating systems as well as operating system updates.

5.1.1 Prototype 1

The initial system developed as a proof-of-concept consisted of multiple sub-systems working in parallel. As seen in the later systems, a web-server, image capture system (through attached camera) and event listener system existed. These were created using the following software solutions:

- Web Server -¿ Apache2, HTTP server hosting software.
- Image Capture -¿ Motion, continuous image capture and serving system using motion detection.

- Event Listener -¿ NodeJS, JavaScript runtime engine (JavaScript software without web browser). Through the use of the Johnny-Five system, NodeJS was capable of communicating with an Arduino running Firmata (communication protocol). Socket listening was accomplished using Socket.IO.

Image streaming was slow and required very low-resolution images (320x240) for real-time transmission over ad-hoc (local) WiFi. Furthermore, the high number of sub-systems and programming languages used was felt to be inefficient. With Apache running web servers across the world, a single-client locally run website did not need such an extensive system.

5.1.2 Prototype 2

The second system developed aimed to minimize the number of subsystems, primarily aiming to develop a single solution using a single programming language. The ability of NodeJS to communicate directly with hardware as well a large, modular, open source library, made it a good candidate for an upgrade control system. The same key subsystems existed, however, these were combined into a single file with no outside software required other than OpenCV for image capture and processing. Client-Server communication was again managed through event-based messages using Socket.IO.

- Web Server -¿ NodeJS, Express module, HTTP web framework.
- Image Capture -¿ NodeJS and OpenCV, an event listener within the NodeJS server listened for client messages requesting image frames, this led to an OpenCV process capturing a new frame as a JPG and returning the file name for serving to the client.
- Event Listener -¿ NodeJS, JavaScript runtime engine (JavaScript software without web browser). No changes from the initial prototype other than merging the system into the server program.

This system proved reliable and much simpler than the previous one. Images were only captured once the client had successfully loaded the previous frame, this limited the frames-per-second to internet and processing speed. However, the camera was reopened for each frame, rather than held open; to solve this the image capture system would need to be run independently (in parallel), and serve on request directly to the user. That said, high-quality streaming was achieved, albeit with noticeable latency.

5.1.3 Final System

The need for an independent image capture system was followed by the introduction of USB2Dynamixel for direct motor control without an Arduino. With a working USB2Dynamixel library already existing in Python, Python was used for this system. Using Python's own ability to listen on web sockets, as well as the default HTML/JavaScript web socket functionality, communication between the client and the motor controller was achieved. This system removed the server as a communication bottleneck, freeing up the server for HTTP serving only. With this accomplished, it was found that Python could serve the web page with no limitations. The Flask framework was used and proved highly successful with a dynamic web

serving ability which allowed for inbuilt camera streaming. This change resulted in near perfect 720p live streaming over a local network.

- Web Server -¿ Python served using the Flask framework (similar to Express used in Prototype 2) with a continuous multi-user ability added through the use of gevent (networking library). The IP address of the server is passed to the client through a Flask variable for use with the event listener.
- Image Capture -¿ Python, Flask and OpenCV, a Flask variable exists within the HTML web page as an image source, on serving by Flask this is continuously updated with binary image data.
- Event Listener -¿ Python and Simple USB2Dynamixel, uses a web socket to receive commands from the client. This is a separate running subsystem on a preset port.

5.2 Hardware

Current dynamic perspective systems rely on the movement of a person head, or eyes, to translate an on-screen 3-dimensional environment. This system relies on the creation and or existence of a computer generated image to navigate within. The computer-generated image is manipulated to replicate natural changes in perception - specifically, viewing angle and distance. While this solution allows for natural viewing of a 3D dimensional object within a 2D dimensional space (such as computer screen), it is only appropriate where the creation or usage of a 3D dimensional image is possible. For use in a real world environment this approach isn't possible without the prior generation of the natural environment by the computer. This approach possibly overcomplicates a process which in humans simply requires minor head movements. A processor intensive solution would be to use multiple cameras to generate a 3D image from the compiled stereo image. This would be a complex system that would need to be able to create a 3D, fully textured (i.e. coloured) replica of the real world at 20-30 frames-per-second, to enable real-time interaction and control. A real-time system would be key to ensuring accurate robot gripper control, any drops in framerate could lead to misjudgments in object handling by the user. A simpler system therefore is using multiple cameras angled towards a common, fixed point in space. Switching between camera inputs, either through face detection, movement detection (i.e. gyroscope) or manually, results in the slight difference in viewing angle expected by side to side head movements when naturally inspecting an object. While visually similar to other stereo camera systems, this system differs in two key areas (other than the addition of a third camera). First of all, no stereo information, such as depth or X is calculated, rather the cameras are used individually to give images with slight differences in viewing angle. Secondly, the cameras are angled inwards towards a common focus point (x cms away), this creates a rotational difference in the given view of a nearby object, rather than simply an offset. INSERT Comparison IMAGE

5.3 Mechanical

As the design and use-case of the robot focused most heavily on low cost, pullies were used to reduce the required torque of motors. MATHS!!

Rather than high cost motors to adjust the height of the arm, a tower was developed to act as a vertical sliding platform for the arm to move across. The use of a double-pulley system as well as two additional single pulleys lowered the force required to pick-up a 2 litre bottle weighing X to X. The double-pulley existed to unravel the two portions of the cable at different speeds. With the one side of the pulley having a radius 2 times that of the other, this ensured that the arm rotated at half the rate of the height change. This created a radial effect, with the robot gripper moving across a radius of X from X.

6 Review

This thesis covers a range of topics, requiring an effective robot arm design, software control system and electrical system to work in parallel while being robust and low cost. The following research explores the possible issues, and required design considerations in developing this system. The robot arm will be controlled manually and largely lack any form of automation or intelligence, due to this focus, research will primarily examine human controlled robotic manipulators rather than autonomous assistive robots.

6.1 Software

While the removal of WebRTC has removed a key cause of software incompatibility, some issues still exist due to the streaming system and the use of touch based controls. As mobile devices operate primarily through touch, conflicts occur on some devices between the browser capturing touch events, and the operating systems attempting to inerpret these also.

System	Compatible
Android	Yes
Windows	Yes
Windows Mobile	Partial
iOS	No

6.2 Hardware

While the multi-camera, "perspective" view system was capable of enabling users to better understand 3-dimensional objects, it was found incompatible with the final robot design. This a result of the complex structure of the developed gripper. To enable a low cost gripper, obtuse mechanical structure such as large springs, and strings to support the pully system, existed within the gripper. These blocked the view of smaller items both prior to, and during collection by the camera system. While effective in placement and viewing angle (71deg) for collecting larger objects, the camera placement also limited navigation, as viewing distance was limited to 70cm (when angled to allow view of the base plate – albeit, with obstructions). By increasing the viewing angle to 85deg in reference to the tower, viewing distance became infinite, however view of the base plate was lost. As a result of these issues, it was found that a new camera system was needed, with the three cameras separated in usage to: navigation, near, and gripper. The navigation camera would have a lower angle (in reference to the mounting plate), allowing greater viewing

distance. The near camera would be used in place of the previous camera system, high above the "wrist" with a viewing angle between the base plate and again 70cm out. Finally, the gripper camera would be placed within the gripper structure, allowing full view of the gripper "finger", and base plate during collection of smaller items. Perpendicular from the base plate, the cameras were initially centered 18.5cm above the wrist center point. For the near camera, an optimal height of 15.5cm and angle of 45deg was chosen to ensure full coverage.

6.3 Mechanical

Upon completion of the robot gripper, the scale and mechanical components of the gripper were found to cause issues with the previously planned camera system. As previously discussed, the camera system was designed to give the user greater viewing angles and 3-dimensional understanding of objects. However, the gripper's components blocked the camera from viewing specific angles such as the final "grabbing" of an object. Furthermore, the low angle of the cameras – aimed towards objects and the gripper – limited the field of view when navigating the robot through an environment. As a result, the camera locations were forced to be re-examined, with an emphasis on allowing clear viewing during all 3 stages of the robot's activities – navigation (infinite viewing distance), grabbing objects (close to medium distance) and manipulating objects (a direct view of the gripper base-plate and "finger").

3D printing allowed for a low cost and continuous development cycle moving through the stages of: design, development, fabrication, and re-development. However, the reliance on 3D printing by the project introduced multiple issues. The most prominent of these were incorrect sizing of components. Measurements were found to be off by up to 0.2mm, with interlocking components suffering primarily. Gaps left to ensure correct fitting of objects were forced to be increased, leading to uncertainties of the extent to whether or not objects would correctly fit. As development continued, the need to reprint large objects due to design changes reflected heavy time investments. Nonethless, the use of 3D printing allowed for parallel development as no manual machining of parts was needed.

7 Discussion

8 Conclusion

References

[1] H. Nguyen, C. Anderson, A. Trevor, A. Jain, Z. Xu, and C. C. Kemp, "El-e: An assistive robot that fetches objects from flat surfaces," in *Robotic helpers, int. conf. on human-robot interaction*, 2008.

[2] K. Tsui, H. Yanco, D. Kontak, and L. Beliveau, "Development and evaluation of a flexible interface for a wheelchair mounted robotic arm," in *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*, ser. HRI '08. New York, NY, USA: ACM, 2008, pp. 105–112. [Online]. Available: http://doi.acm.org/10.1145/1349822.1349837

[3] K. M. Tsui and H. A. Yanco, "Simplifying wheelchair mounted robotic arm control with a visual interface." in *AAAI Spring Symposium: Multidisciplinary Collaboration for Socially Assistive Robotics*, 2007, pp. 97–102.

[4] K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, and J. Wiegley, "Desktop teleoperation via the world wide web," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 1. IEEE, 1995, pp. 654–659.