ESE-519: Real Time Embedded Systems

Project Content Reactive Toys

Sarath Vadakkepat, Stuart Helgeson & Zairah Mustahsan

1. Project Title: Content Reactive Toys

2. Team:

A. Stuart (Lead for Hardware (component selection, designing circuitry, mechanical design))
B. Sarath Vadakkepat (Lead for Communication)
C. Zairah Mustahsan (Lead for Media processing)

3. Motivation:

Content reactive toys enable kids to have a much more immersible experience while watching their favourite action anime/cartoon/movies. Content reactive humanoid toys provide a new dimension to toys which can be commonly christened "Smart Toys". Content reactive toys react and enact out their virtual self in an anime/cartoon/movies. It would be desirable to design such toys that successfully enact a sequence or a set of actions in sync with a media program with a very low latency.

4. Goal:

The outcome of the project would be create a prototype of an action humanoid toy figure which can imitate few of the motions synchronized with a media file. Various motions could include movement of hands and legs, vibration and sound sensors and speakers and limited toy figure movement.

5. Methodology:

1. Design the complete blueprint of the system on paper. Identify the key components required.
2. Design and build a body enclosure for the hardware in the form of humanoid toy figure.
3. Embed the hardware into the built body to form a prototype of the humanoid
4. Set up a communication medium between the hardware and software components.
5. Develop a Web Application [HTML5] to deliver media content in the form of videos based on the response from the child.
6. Detect the response of the child at various set points in the videos [this is done by detecting the movement of the toy in different ways, using a 9 DOF IMU].
7. The data from the toy is sent over to the server which determines if the child's response is right or wrong.

8. Once the server has estimated if the response is right or wrong, it communicates with the Web Application module which displays a new video.
9. Add exciting and interactive media content such that children enjoy the videos along with playing with the toy. For example, if the server predicts that the child is not responding and he/she has remained idle for a long time the media content displayed will have a repetitive pattern which will in turn force the child to play with the toy, to be able to watch new and exciting videos.
10. Test the complete system with media file and hardware components on the humanoid.

6. Architecture:

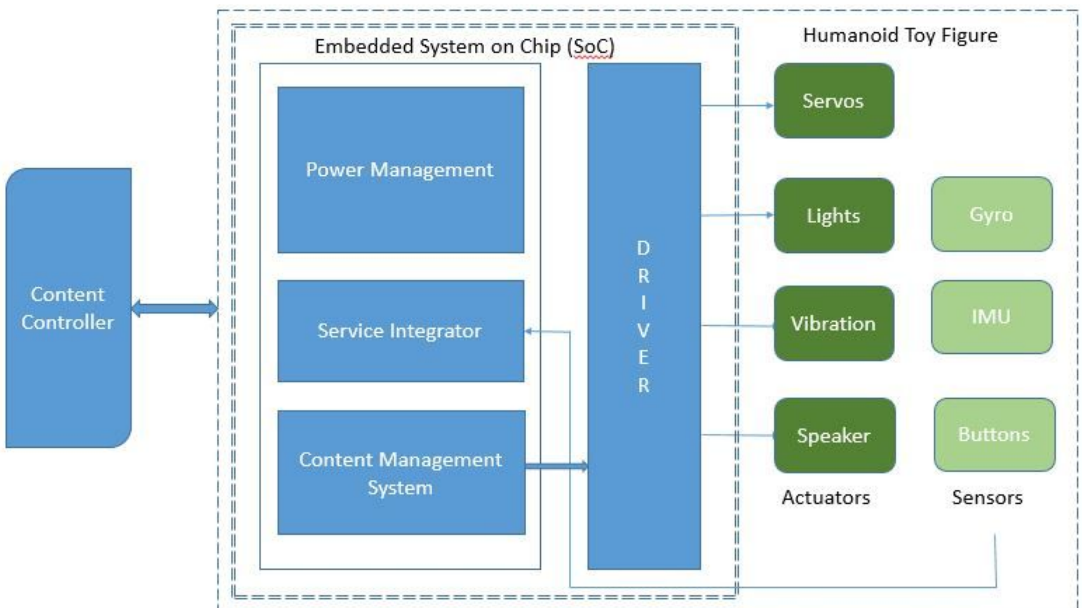


Fig. System Architecture

7. Project Components:

<u>Hardware :</u>
Raspberry Pi as the microcontroller
Servo/Speaker
Vibration Motors/LED lights/Capacitors/Resistors/Inductors/Switching Voltage Regulator

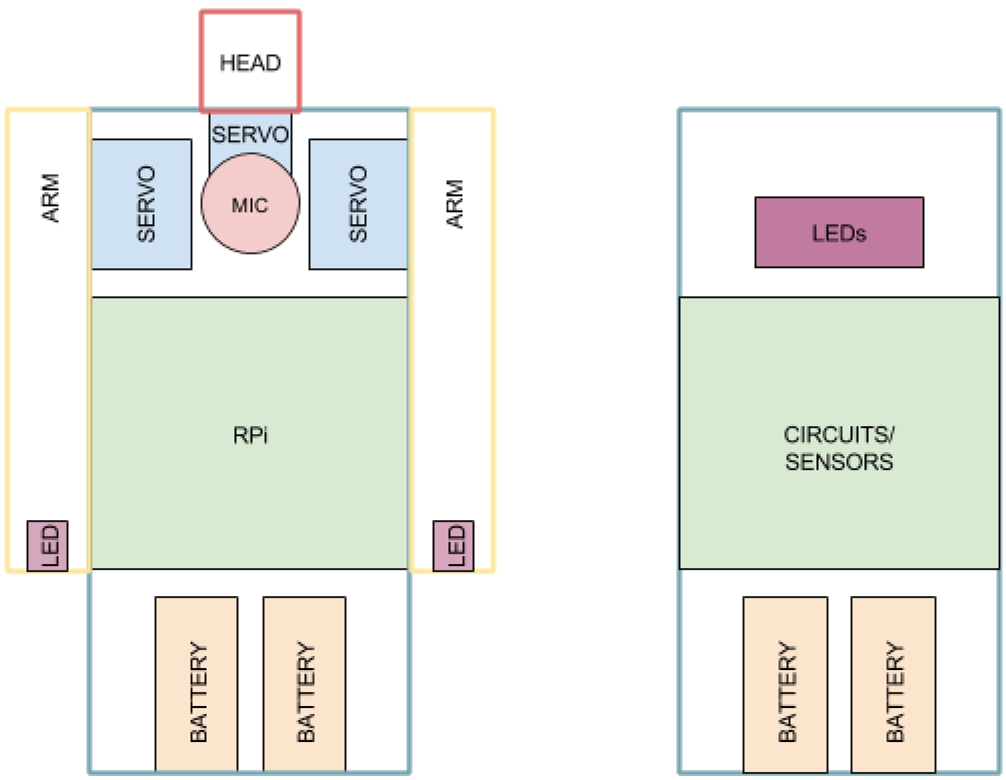
<u>Software :</u>
Python based TCP/IP communication protocol
Data file containing content reaction data required.
Timing program to sync the media file and the content reactive data sent over communication network.

7.1 Hardware Effort/Mechanical Effort:

- Sensor interfacing [9 DOF IMU with Raspberry Pi]
- Compact circuitry design
- Soldering
- Designing the toy - Laser cutting, 3-D printing.
- Placement of IMU and other components on the toy.

Mechanical:

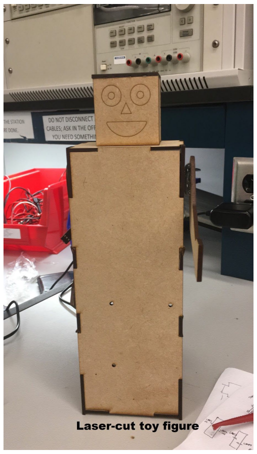
The toy was designed so that various humanoid shapes could be made to fit the hardware. The architecture of the layout is described below.

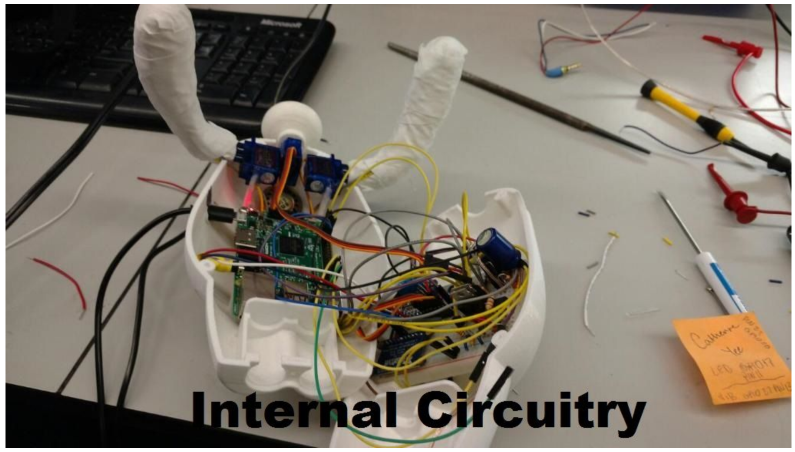


BACK FRONT

With this design, toy designers have the flexibility to modify the shape of the toy to fit their desired designs.

Our initial design was made up of laser cut 2D parts to rapidly test the layout of the components. It helped visualize the spacing of parts. After learning from the initial design, our final design modeled after Big Hero 6's Baymax. All the custom parts were designed in SolidWorks. The parts were made utilizing SolidWorks' surface modeling to create the complex organic shapes (all CAD files are on Github). The custom parts were 3D printed in order to rapidly manufacture and test the designs.





Electronics:

There are four major circuits: power management, sound amplifier circuit, 9DOF IMU circuit, vibration motor, and LED control circuits. Our device had high power demands because we chose to use a Raspberry Pi as our microcontroller. In order to supply enough current and meet our voltage needs, we settled on single cell, rechargeable lipo batteries and a boost circuit to supply a reliable 5V. We found that it was quite difficult to power everything at once as the device could draw over 1A at times. We did not foresee the amount of power the Raspberry Pi would need while using wireless. Nonetheless, we used a MC34063A in a boost configuration to step up 3.3V to 5V. This supplied all the peripherals: sound amplifier IC, IMU IC, servo motors, and LEDs.

To control the servo motors position, the Raspberry Pi sends a PWM signal. It also sends the audio signal to the sound amplification circuit. The sound amplifier circuit drives the internal speaker. The IMU communicates to the Raspberry Pi through I2C protocol. LEDs are controlled by high or low signal from the Raspberry Pi to a power transistor. The vibration motor has a similiar setup.

7.2 Software Effort:

- Binary Tree inspired recursive media display and reaction algorithm
- Websockets using Tornado framework
- I2C interfacing of IMU 9DOF to Raspberry Pi 3
- Wifi Integration for raspberry pi
- Web Sockets
- Multithreaded Python based service integrator
- Integrated Python based driver
- HTML 5 Video player
- AWS S3 to host and stream video.

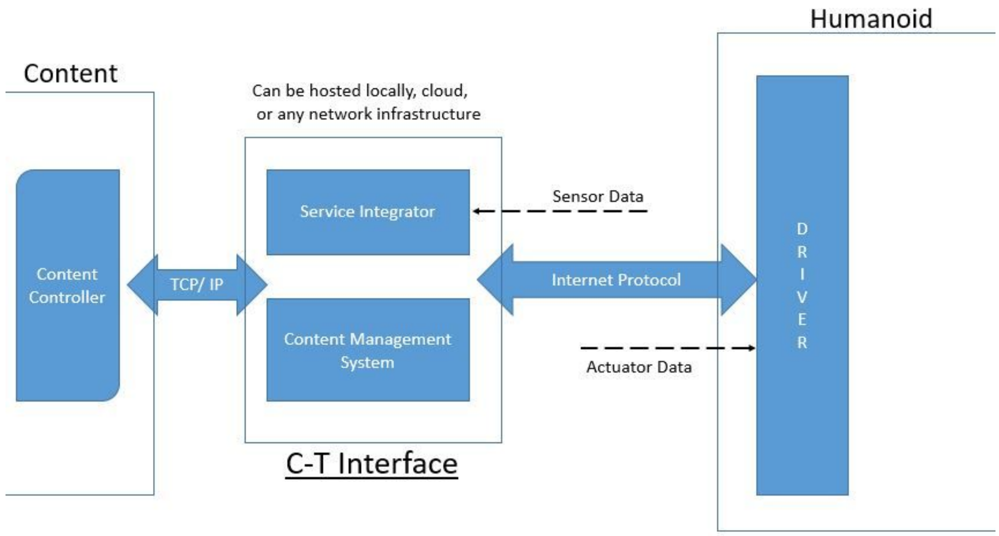


Fig. Software Architecture

8. Testing and Evaluation:

Testing:

- Tested the robustness of the device by performing various action sequences.
- User input from the toy
- Tested our recursive media display algorithm to ensure that our system always displays the media content based on how child responds.
- Stress tested the Python Server and the HTML5 client web APP by sending in concurrent requests.
- Ensured the hardware actions are visible for a given media file with responses to the media file in minimal delay. (Reach Goal)

Evaluation:

- Self tested the system in a closed environment
- We asked our fellow classmates to play with the toy, and obtained feedback from them.
- Implemented few modifications to the system based on user feedback the baseline demo.

9. Project Deliverables:

Baseline 1: Design the complete blueprint of the system on paper. Identify the key components required. Divide the system into hardware and software components and identify the features to be implemented

Baseline 2: Set up a communication medium between the hardware and software. Process a media file to accumulate content reactive data. Make the hardware components respond to different inputs.

Baseline 3: Test the system till this step on a raw scale

Reach Goals:
<u>Reach</u> <u>1:</u> Design and build a body enclosure for the hardware in the form of humanoid toy figure.
<u>Reach 2:</u> Embed the hardware into the built body to form a prototype of the humanoid
<u>Reach</u> <u>3:</u> Visual experience of complete system with media file and hardware on the humanoid.

Following are the actuation points in the toy -

1. Movable head
2. Hands - up and down movement
3. Lights and Sound

Following are the features -

1. Imitation of the content
2. Enable the child to respond to some instructions like picking up the toy, that changes the content accordingly.
3. Enable the toy to vocally articulate some instructions for the child.
4. Reactive content as reflection to child's interaction with toy.
5. Attracting the kid to play with the toy.

10: Hardware Communication:

Content Controller :

This module is developed to stream content from any services such as cloud or hosted locally into a HTML5 video player along with metadata for the video that enables the toy to reflectively enact of the content of the toy proactively.

Service Integrator :

This module is designed to collect all the different kinds of data that the application receives so that it can be filtered and smoothed and validated to only send the right and correct data to the toy.
This is implemented as multi threaded python based program using concepts of microservice architecture of software development (SoA).

Content Management System:

The content management systems brings together the video, the metadata for the video, the actuator values and the inputs into the toy from the sensors into one common decisive place which will help the system to understand the next sequence that have to be shown to the kid based on a complex decisive making algorithm that considers the instantaneous state of the system based on the aforesaid parameters and outputs a single video content file name to be played next.

We also designed and implemented code that can generate metadata for the given video and output the result into a text format which consists of timestamped values of actions to be needed.

11. Links:

Github:https://github.com/ese-519/Content-Reactive-Humanoid-Toy-Figures
Devpost: https://devpost.com/software/content-reactive-humanoid-toy-figures

12. Conclusion:

In the yester years we have seen television and toys as 2 separate entities and the current trend is to see them unified into one. A single entity with a child that can respond and enact out to the child to encourage the child to learn and be involved in a new TV watching experience is what we tried to demonstrate through this project. We tried to bridge this gap by building a closed loop system where in there is interaction between the child and our system back & forth at all times. There is always scope for improvement and we can make our system better by improving the functionality of our different modules and by providing more depth in the content video hierarchy tree structure giving many more paths for the child to cover and engage themselves.