Tracie Report

Team 22 [*]

November 27, 2014

[*]Members: Mitchell Kember, Do Gyun Kim, Charles Bai, Xiao-Yang (Michael) Min, Min Suk Kim, Renato Zveibil, Leong Si

Table of Contents

1 Executive Summary

"Tracie" utilizes the Scribbler Bot and the Fluke 2 Board to trace out the drawable input transferred from the user interface, via a web app, through linear movement and precise rotations. Programmed in JavaScript, the client, which runs in the browser, is the ultimate form of communication between the user and the robot. The client provides the user the ability to start and stop the program immediately, and allows the user to modify the data values of the Controller, which alters the distance and time conversion and rotation to time conversion for op-

timization purposes. The user interface, integrated within the client, allows the user to input a set of points and provides backwards and clearing functionality. When the program runs, the client communicates with the Server and Controller, both simultaneously running in the terminal, to start up the Tracie script and retrieves information influencing the robot's performance, such as the speed. The Tracie program is coded in Python using the Myro library [1], a Python-written framework used for programming robots. The Tracie program supplies the artificial intelligence and program logic to manipulate the Scribbler Bot's movement.

This report contains a Work Breakdown Structure (Figure 1), a PERT Network Diagram (Figure 2), and a Gantt Chart (Figure 3). They provide a strong foundation in optimizing the time spent on each activity through an organized schedule. Additionally, this report includes a requirements document, which identifies the personnel in the group and their respective roles and contributions, and lists the functional and non-functional requirements in order to progress and complete the project. Furthermore, a computation decision making (CDM) section will also be included. CDM is essentially the description of how the problems in the project are being analyzed, as well as the number and description of the criteria and alternatives found. The project retrospective section will include what worked well and what did not in regards to the supplied resources, (such as the Scribbler Bot or Fluke 2 Board), team organization, and the development process.

2 Requirements

2.1 Roles

Mitchell Kember: Project leader, coordinator, and organizer. Mitchell set up Trello boards [2] to keep track of progress and objectives for the project. Additionally, he set up the client and server [3][4] on which Tracie is run on and contributed the majority of the code.

Do Gyun (Justin) Kim: User interface designer. Justin created the user interface and its functions with JavaScript.

Charles Bai: Algorithm developer and reporter. Charles created algorithms for rotation to time conversion and distance calculations. He also wrote the Executive Summary, designed the Planning Diagrams, and worked on the Requirments part of the report.

Xiao-Yang (Michael) Min: Manager. Michael verified when objectives should be completed in order for the project to be completed on time.

Leong Si: Reporter and report designer. Leong was responsible for the publication of the report. He also worked on the Requirements, Computation Decision Making, Project Retrospective part of the report.

2.2 Functional Requirements

Server: A local host server which allows communication between the user interface on a computer and the Scribbler Bot.

Controller: A program written in Python to control the robot via the server. It should be capable of controlling the robot's movement with different input values.

Math Utility Calculations are needed to be done for the robot to move in the correct speed, angle, and distance. Those calculations will be used to implement the movement algorithm once the calculations satisfy the expected result.

Movement Algorithm Using Math Utility, the algorithm should translate the desired shapes or drawings into a sequence of robot movements.

User Interface The user interface should allow the user to select or create drawings within itself.

Graphical Design The layout of the cilent, which contains the Controller and user interface, should be clean and readable. It should display the current status of the robot, allow the user to change values of the Controller, and have buttons for functions like reset or stop.

2.3 Non-functional Requirements

Constraints

Time

The project must be finished by November 20^{th}.

Money

The team would not receive any payment during or upon finishing the project. Also, the team did not plan to spend extra money on this project, so resources were limited.

DC Motor

The DC motor of the Scribbler Bot made it very difficult to do fluent movement, especially turning. Since the Scribbler Bot was the only robot allowed, many more calculations and considerations were required.

Bluetooth

The range of the bluetooth device in the Fluke 2 Board was limited. It also had a noticeable delay when performing real-time tasks. Different methods were going to be used to bypass these problems, for example, a web client to extend the range.

Criteria

- **Accuracy of the shapes**
- **Time required to trace**
- **Feasibility**
- **Flexibility**

Details of the criteria will be discussed on the Computation Decision Making section.

3 Diagrams

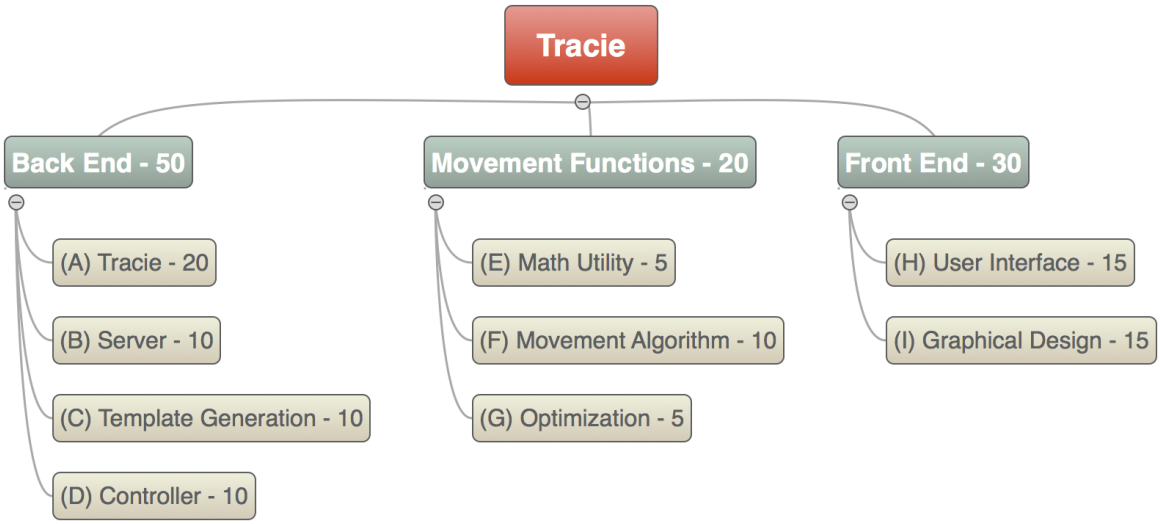

Figure 1: Work Breakdown Structure of Tracie

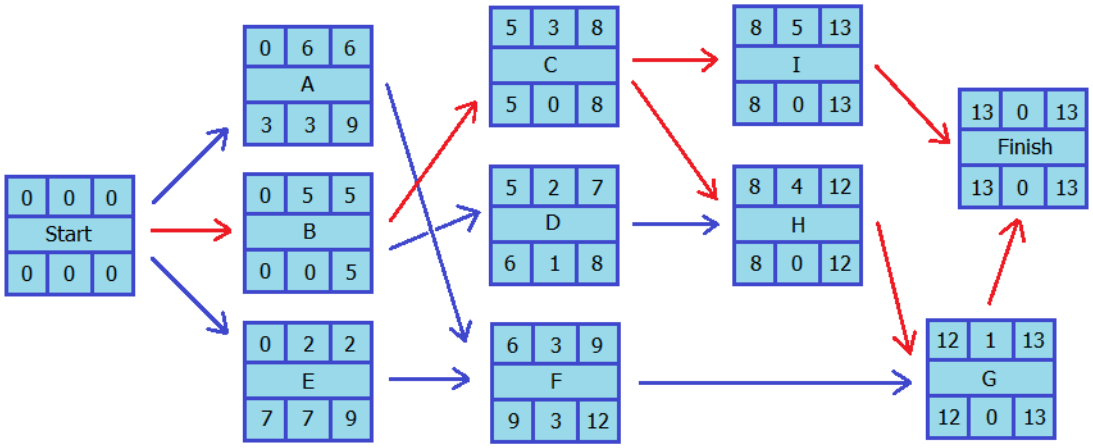

Figure 2: PERT diagram. Critical paths: {B,C,H,G}, {B,C,I}.

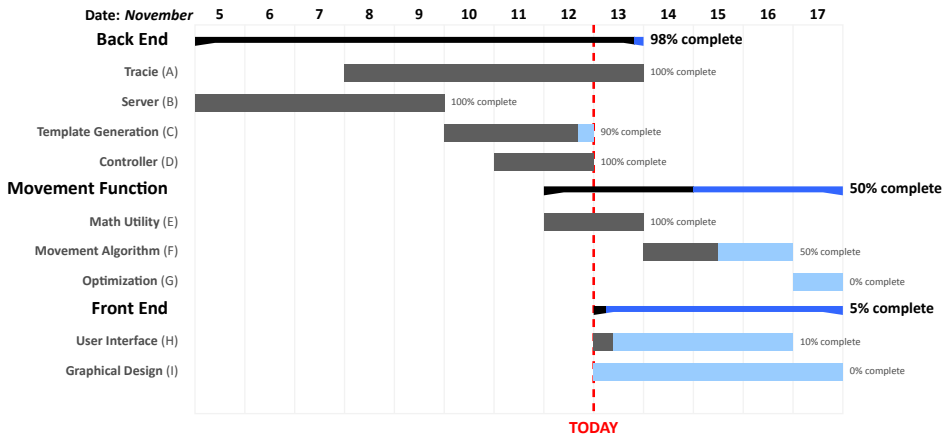

Figure 3: Tracie's Gantt Chart on Nov. 13, 2014 [5]

4 Computation Decision Making

After reading the proposals, the team came to a conclusion that the drawing feature of the Scribbler Bot should be utilized. The original plan was to allow a user to input a drawing from a computer, then the drawing would be traced by the Scribbler Bot on a piece of paper.

There were many criteria needed to be considered before selecting the most appropriate drawing method.

1. **Accuracy**

   The drawing created by the robot must follow very closely relative to the drawing on the screen.

2. **Time**

   The robot should take minimal time to trace out the drawing.

3. **Feasibility**

Some of the members in the team had little to no experience in programming, especially in Python. In order to share the workload equally, the project should not be too difficult for anyone.

4. **Flexibility**

   The robot should create different drawings based on the user's choice.

With the criteria listed above, 3 of the suggested methods were chosen to be computated with CDM.

1) Fixed Shapes

Allow the user to choose a shape from the user interface. After they have selected their shape, the robot will then draw it.

2) Pixel Based

The users will create a pixel based drawing on the user interface, then the robot will recreate the drawing on a piece of paper.

3) Point to Point

The user will insert "points" on the user interface. Each point is connected directly to the point placed before it. The robot will then follow the points in order, thus create a drawing.

Table 1: CDM of Tracie

Criteria	w_i (%)	Fixed Shapes			Pixel Based			Point to Point		
		C_{i1}	r_{i1}	S_{i1}	C_{i2}	r_{i2}	S_{i2}	C_{i3}	r_{i3}	S_{i3}
Accuracy	30	10.0	1.0	0.30	9.0	0.9	0.27	8.0	0.8	0.24
Time	10	10.0	1.0	0.10	3.0	0.3	0.03	7.0	0.7	0.07
Feasibility	30	10.0	1.0	0.30	5.0	0.5	0.15	9.0	0.9	0.27
Flexibility	30	1.0	0.1	0.03	10.0	1.0	0.30	9.0	0.9	0.27
				0.73			0.75			0.85

From Table 1 the results showed that the Point to Point method satisfied majority of the criteria. Therefore, it became the most suitable method of tracing for the project.

5 Project Retrospective

5.1 Resources

During the production of Tracie, the team faced different kinds of problems, especially the ones involving hardware. The robot provided, the Scribbler Bot, was not equipped with well-functioning hardware. The sensoring devices of the robot could have been utilized for the project, but the team decided against it since the devices were not reliable. Instead, the most consistent parts of the robot were used, the motor and the drawing “function”.

The Fluke 2 Board was also a supplied resource. Similar to the robot, the sensoring devices could not provide the accuracy needed. Also, as discussed in Section 2.3, the bluetooth connection was a major problem throughout the project.

The Myro library was used in the code to access parts of the Scribbler Bot. It contained all the functions needed to control or receive information from the robot. It made coding for the project very simple. However, the only problem the team struggled with was the precision of the robot's rotation. The functions in the library allowed the robot to turn or rotate one of its wheels for a specific amount of time, but there was no function that allowed the robot to rotate a specific angle.

5.2 Project Organization

The final product, Tracie, was consistent and accurate. The project served as a benchmark in order to learn and practise the development of software. Futhermore, the challenge of hardware constraints served as a learning experience about the practicality of software.

The development process of the project went well. All the components needed to progress were finished ahead of schedule. Thanks to Mitchell, who set up the GitHub repository and the Trello boards, the group was able to clearly identify the critical path and important deadlines.

Although the project went very smoothly, there were conflicts within the organization of the group. There were difficulties in managing the group schedule. The result was an unbalanced workload distribution within the team. While some of the members attended all meetings and participated in all aspects of the project, others did not contribute.

5.3 Solutions and Alternatives

The main issue within the group was an organizational one. To resolve issues similar to this in future cases would involve setting up mandatory meetings for group members where everyone must show up and work together on the project. Also, planning a schedule at the beginning of the project can provide insight on when the meetings are, so that individuals have an adequate amount of time to adjust their schedules.

6 Reference

[1] Institute for Personal Robots in Education. (2013, April 24). *Myro Reference Manual* [Online]. Available: http://wiki.roboteducation.org/Myro_Reference_Manual

[2] M. Kember. (2014, Oct 27). *Tracie* [Online]. Available: http://www.trello.com/b/iJbcFOOv/tracie#

[3] D. Bilenko. *Gevent: A coroutine-based network library for Python* [Online]. Available: http://www.gevent.org

[4] M. Hellkamp. (2014, Nov 25). *Primer to Asynchronous Applications* [Online]. Available: http://www.bottlepy.org/docs/dev/async.html

[5] W. Skala. (2013, June 1). *Drawing Gantt Charts in LaTeX with TikZ* [Online]. Available: http://mirror.math.ku.edu/tex-archive/graphics/pgf/contrib/pgfgantt/pgfgantt.pdf