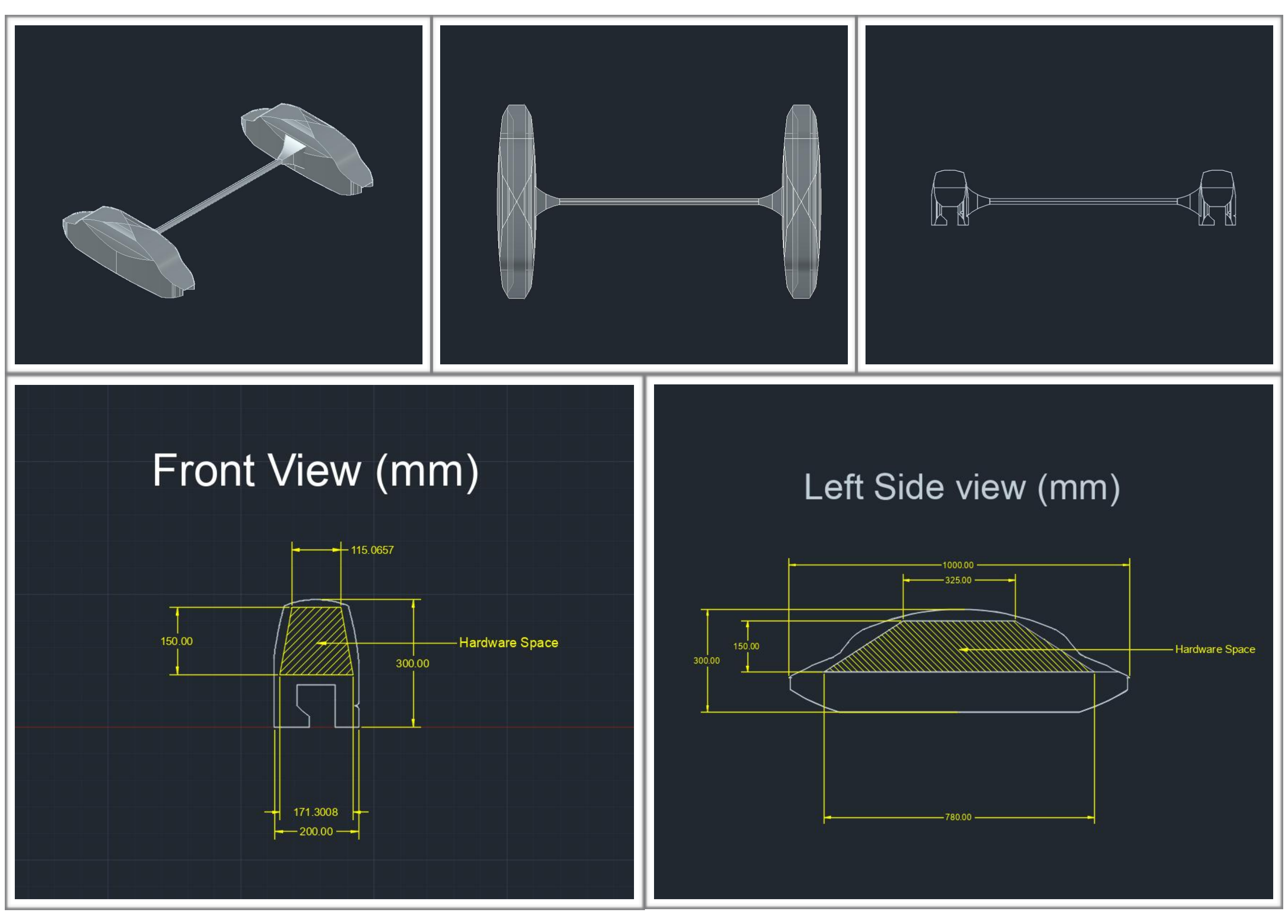
Software Development

Project Outline for Sacha



Hello Sacha, our team successfully was able to get all four people for the completion of this project, which is awesome! That being said, there is a lot of room for confusion, misinterpretation, and error, since all four of us are trying to tackle one idea! Therefore, for organizational purposes, I have attached a table below, outlining what you should complete in order for this hack to go smoothly.

(Please note, I am not trying to micromanage the team and you are free to make changes! However I am doing this so there is consistency between the hardware and the software components of this project. Have fun!)

Our team's hardware designer, Aaron, will design the device that will,

- Detect objects in between the rails
- Detect any objects *on* the rail within 50m in front of the device
- Use two infrared sensors to sense heat coming from the left and right sides of the device (Used to detect animals/people near the tracks)
- Be equipped with a camera, on both devices, facing both rails in order to detect metal fatigue. An ultraviolet light will be turned on in order for the metal fatigue to be visible.
- Detect if the device is moving or not. If not, there might be an obstacle in the device's way preventing it from moving and posing a hazard to the train.
- Be equipped with a transceiver that can communicate with other transceivers within 5km.
- Be equipped with a bright flashing white light to warn others of the device's presence.
- Have a bright flashing red light to indicate an emergency.
- Be able to turn on/off via the micro-controller. The Raspberry Pi itself does not turn off, just the other components of the RODD (Railway Obstacle Detection Device). A physical switch must be used to turn the Pi off.

In addition, Aaron will provide you with a **hardware library**, which will enable you to program the device's components. (e.g. move() method will move the device). The documentation for his library is at the bottom of this document.

Please read the table below beforehand, so that when the hackathon begins, we can hit the ground running!

Tasks	Description
In Java, write the software for the RODD (Railway Obstacle Detection Device). The software will include Aaron's hardware library to give you access to the device's hardware. The software criteria is outlined in the description.	• The software must: • Stay *x* metres ahead of the train. The distance between the device and the train will be given to you as an argument for your program: int DIST_FROM_TRAIN. Note: Use Aaron's move() function to move. • Check for the following every 200 ms. • Any objects in between the rails. • Any heat "blobs" detected by the two infrared cameras. Note: you might need some external library to recognize these patters, or ML. • Any obstacle within 50m in front of the device on the rails • Detect any metal fractures on the rail images that are captured using the read_rail_cameras() method. Any metal fractures under the UV light will look like a bright purple hairline crack. See Google for images. ML or a library might be needed to detect these cracks. • If the device is trying to move, but has no speed, meaning it is stuck. • If any of the above are detected, trigger a hazard function that sends a signal from the onboard transceiver and turns on the hazard light. • Power on and off by an input signal from the transceiver.
Add the files to our GitHub repository	

Aaron's Hardware Library Documentation

Function / Method	Description	Return value
move (int speed)	• moves the RODD • Accepts the speed in km/h as an argument (negative if reversing)	*Signed Integer* • Returns the current speed of the device as a signed integer. (negative if reversing) • Returns infinity (or negative infinity) if the device does not accelerate (meaning it's stuck) for more than 10 seconds.
object_between_tracks ()	Detects wether or not there are foreign objects in between the rails.	*Boolean* Returns true or false wether or not an object between the tracks is detected.
obstacle_on_rails ()	Detects if there is an object on the rails within 50m in front of the device.	*Boolean* Returns true or false wether an object was detected.
read_infrared_cameras ()	Reads the two onboard infrared cameras in order to detect animals near the train tracks. (For details on this function and the data types and libraries used for it, please correspond with Aaron.)	*Array of a data type that is TBD* Returns the raw data of both left and right infrared cameras in an array with two keys: "left" and "right"
read_rail_cameras ()	Reads the two onboard rail-facing cameras (closeup of the rail below the device). With the help of the UV light, any fractures in the metal can be visible through the camera. (Again, please correspond with Aaron for the details of the method)	*Array of a data type that is TBD* Returns the raw data of both left and right rail cameras in an array with two keys: "left" and "right"
transceiver_in ()	Loads serial data from the onboard transceiver's buffer. This will be the main way of receiving data from the locomotive.	*String* Returns the decoded message from the transceiver.
transceiver_out (string data)	Send data out of the onboard transceiver. This will be the main way of sending data to the locomotive.	*Boolean* Return true or false wether or not the data was successfully sent.
power (int mode = 2)	Turns the RODD on or off, depending on the "mode" parameter: 0 = off 1 = on 2 = toggle.	Returns the new state of the device (false for off, true for on).
light (int light, int mode)	Turns a light on/off. "light" parameter selects the light to turn on/off: 0 = Flashing white light (on light) 1 = UV light for rail camera 2 = Emergency indicator light. The "mode" parameter determines what to do with the light 0 = off 1 = on 2 = toggle	*Boolean* Returns the new state of the light: true for on, false for off.