ShARC NXP Cup Car - Interim Report

Andrew Huff, David Chung and Hamish Sams

***Abstract*—This report is a summary of current progress of the Sheffield Autonomous Racing Car (ShARC) covering how the aims and objectives have changed over time and the associated project risks. A work program is also formulated creating a general plan of who and what needs be completed with an associated gantt chart.**

CONTENTS

I Project Review 1
- I-A Aims and Objectives 1
  - I-A1 Revised Primary Objectives 1
  - I-A2 Revised Secondary Objectives 2
- I-B Specifications 2
  - I-B1 Mechanical and Power 2
  - I-B2 Steering 2
  - I-B3 Computer vision . . 2
- I-C Risk register 2

II Revised Work program 4
- II-A Andrew 4
- II-B David 4
- II-C Hamish 4
- II-D Gantt chart 4

III Project Progress 6
- III-A Steering 6
  - III-A1 Component Choice 6
  - III-A2 Algorithm 6
  - III-A3 Mapping 6
- III-B Hardware 6
  - III-B1 Chassis 6
  - III-B2 Battery 6
  - III-B3 Motor and Gearing 7
  - III-B4 Motor Controller . . 9
  - III-B5 Miscellaneous electronics 9
- III-C Computer vision 9
  - III-C1 Coral Dev Board . 10

References 10

I. PROJECT REVIEW

A. Aims and Objectives

The overall project aims and objectives were set in [1]. The project aims remain unchanged other than the new deadline to have the vehicle ready for April, when the UK qualifiers will be held, rather the final competition in May, as good performance is a requirement of qualification.

Since the project aims to take part in a competition, the objectives remain broadly similar as these are based on the requirements and rules set by NXP. As expected at this stage of the project, only research and purchasing of major hardware has been fully completed, however many more are in progress. Due to the limited time left before the competition qualifiers, several of the secondary objectives have been relegated to future developments and are unlikely to be implemented before the competition. These are:

- A full computer vision system will only be developed to detect the obstacle for the obstacle avoidance discipline. The bulk of the steering, intersection and speed limit detection will be offloaded to the recently purchased Pixy2 camera which offers pre-developed computer vision algorithms.
- Torque vectoring will not be possible in the time available due to the complexities of physically building the hardware as well as the complex control systems needed.

1) Revised Primary Objectives:

- Implement a reliable computer vision based steering control and speed setting system to follow the track and deal with intersections/chicanes to complete the main event
- Develop object detection system to avoid a partial track obstruction by a hard-to-see object to complete the object avoidance discipline
- Develop an object detection system to emergency brake and avoid a full track

obstruction by an easy-to-see object to complete the emergency brake discipline
- Implement a speed limit sign detection system to complete the speed limit detection discipline

2) Revised Secondary Objectives:

- Develop a racing line algorithm on top of computer vision to improve lap times
- Refine the steering algorithm for faster lap times for the 'figure of eight' discipline

B. Specifications

From the revised aims and objectives, the final specification can be made for our design. This specification allows us to ensure our components, if exceeding these requirements, can always achieve the minimum performance needed to race using the worst-case driving strategy:

1) Mechanical and Power:

- Chassis: 3Racing Sakura M4 four-wheel-drive
- Motor: Turnigy TrackStar 21.5-turn, 1855Kv, 22A, 8.4V
- The final drive ratio is 5.7 using an 80-tooth spur and 28-tooth pinion
- The car can exceed an acceleration of 8.82ms^{-2} up to a speed of at least 6.6ms^{-1}
- Battery: 3s Sony VTC5A, 30A continuous, 2600mAh capacity
- Motor controller: Roboteq SBL1360, 6 step commutation, 20A continuous, 9V-60V, regenerative braking, closed loop speed and torque control, RS232 and USB
- Regulator: Matek Systems Duo UBEC, dual $5V/5-12V$ output, 4A per output, switching

2) Steering:

- Servo: Trackstar TS-920 Digital 25T
- The steering motor is to have a torque of at least 12 kg/cm (7.4V), based on the example build by the manufacturer of our chosen chassis. Exceeding this further improves the steering stability.
- The rotation speed of the servo must meet or exceed 0.07s / 60 degs to meet the manufacturer's recommended component's performance
- The steering motor must be digital to allow for maximum holding torque, smooth output, and significantly faster response time as opposed to analogue.
- The servo must be a standard size category to fit the chassis's servo fitting.

3) Computer vision:

- Camera must be able to maintain a viewing angle of at least 90 degrees to keep track corners in view.
- Camera must maintain a framerate of at least 30 frames per second to ensure controller input is real-time.
- Vision must be capable of identifying two 20mm width black lines which represent the track edges.
- Vision must be capable of identifying a coloured low power laser on a white 20x20x20cm box
- Vision must be capable of identifying a 3 stripe and 4 stripe pattern, indicating the two different speed zones
- Vision must be capable of identifying a 20x20x60cm black box, indicating the emergency brake line.

C. Risk register

Almost all hazards of the project are the same as defined previously, the causes and therefore measures to reduce this risk have, however, changed in turn changing the risk rating. The largest risks currently are time limit issues given the short duration of time and vast amounts of work due before the qualifiers and final competition in April and May respectively. The other largest risk is software issues given the reliance on a single camera not yet programmed, this is therefore the current biggest priority in the project as all other deliverables rely upon this. Helpful resources are available at the NXP gitbook which should help achieve this deadline found in the gantt chart below.

Hazard summary	**Existing Measures**	**Likelihood**	**Severity**	**Risk Rating**	**Additional Measures**	**Residual Likelihood**	**Residual Severity**	**Residual risk rating**
Data Loss	Software created for the RC car is using cloud based version control (gitHub) allowing for simultaneous code writing and secure cloud storage.	1	4	4	N/A	N/A	N/A	N/A
Over Budget	No more purchases are planned in the future and all components are ordered, the only need for components will come if a device is broken in use and as such caution should be taken handling and using components especially those expensive and critical to operation.	2	2	4	N/A	N/A	N/A	N/A
Time Limits	Following the generated Gantt chart, the project will meet all deadlines and thus following this schedule is important and any deviations must be planned cautiously.	3	3	9	The high risk of running overdue should be mitigated as much as possible by making sure progress is being made in each area with all members knowing exactly what must be done, this should be explicit in meetings.	2	2	4
Rule Breakage	All parts have been bought within regulations and will be written into a log to meet the rules. The batteries used must have extra justification given the vague text in the rules later explained in email.	2	4	8	N/A	N/A	N/A	N/A
Car Breakage	For testing the car a Bluetooth kill switch is currently available allowing for remote shutdown where the motor controller automatically brake.	1	4	4	N/A	N/A	N/A	N/A
Lost time injury	Risk assessments should be read and followed to prevent injury that may lead to lost time injuries. Lost time on such a short and demanding time scale can lead to large problems.	2	3	6	N/A	N/A	N/A	N/A
Hardware issues	A static friendly environment should be used when handling sensitive electronics and resources such as the NXP Github should be used when dealing with recommended hardware.	3	2	6	N/A	N/A	N/A	N/A
Software issues	Extensive testing should be used alongside a modular style for future code to allow easy merging for different races and for ease of understanding, debugging and generally good code. The NXP gitbook should be used as a reliable source of recommended hardware code.	3	3	9	N/A	N/A	N/A	N/A

Risk Rating Reference

	Likelihood				
Severity	**1**	**2**	**3**	**4**	**5**
1	1	2	3	4	5
2	2	4	6	8	10
3	3	6	9	12	15
4	4	8	12	16	20
5	5	10	15	20	25

Risk Rating	**Explanation**
1-5	No additional measures needed but can be implemented to reduce further risk.
6-12	Decide whether further measures need to implemented to lower risk rating.
15-25	Stop the corresponding task(s) immediately and seek to reduce risk.

TABLE I: Revised Risk Register

II. REVISED WORK PROGRAM

A. *Andrew*

- Basic steering and speed with Pixy2 camera
  - Set up Pixy2 with PC software
  - Establish communication between Pixy2 and FRDM-k64f
  - Form vector of track based on edges
  - Use vector angle/length for input to proportional steering
  - Use vector angle/length for setting speed setpoint
  - Test above system on example track
- Intersection and chicane handling with Pixy2
  - Test how system performs without explicit handling
  - If necessary, detect chicane and intersections
  - Implement algorithm to avoid steering in above scenarios
- Mount 18650 cell holders to base plate of chassis

B. *David*

- Detection Method
  - Compare CoralDev and pixy2 for image processing and detection
- White Object Detection
  - Research methods (e.g lasers) to highlight white box
  - Implement colour recognition software
  - InvestigateInvestigate avoidance strategies
  - Test system on example track
  - Add to hot-swap system
- Speed Zone
  - Implement sign detection software via pixy2
  - TeachTeach pixy2 speed zone signs
  - Implement speed up and slow down responses to relevant sign
  - Test system on example track
  - Add to hot-swap system
- Emergency Brake
  - Determine between sonar and computer vision methods for detecting box
  - (Sonar) purchase hardware
    - ∗ Establish communication between device and FRDM-k64f
    - ∗ Use measured distance as input for open loop braking controller
    - ∗ Determine braking distance for RC car
    - ∗ Use braking distances as the threshold for braking controller
    - ∗ Test system on example braking track
    - ∗ Add to hot-swap system
  - (Computer Vision) Implement colour recognition software
    - ∗ Investigate object differentiation between black box and track edges
    - ∗ Use black box size as input for braking controller
    - ∗ Determine braking distance for RC car
    - ∗ Link object size to distance to find threshold for braking controller
    - ∗ Test system on example braking track
    - ∗ Add to hot-swap system

C. *Hamish*

- Hotswapping system
  - Algorithm to allow switching of software components for different races
  - Hardware input for what race the car is running to manually enter
  - Connectors for any race specific hardware
- Electronics top board
  - Design component layout of top board
  - Create CAD design of top board
  - Laser cut/create top board
- Create and mount camera holder
- Create hardware log as requested in the NXP cup rules

D. *Gantt chart*

This project is based around meeting the qualifiers deadline where the car must be ready to compete in all races and thus must have all hardware and software ready by this date. The gantt chart seen in Fig. 1 shows these deadlines and the work associated planned. Deadlines after this are for university deliverables and thus no hardware changes need to be made and focus should be on quantifying the racing car meeting the specifications set and how well this is done and presenting this in the best manor. The gantt chart shows the amount of parallel work needing to be completed where the qualifiers, presentation and final submission work need be done at the same time and requires strong time and work load management.

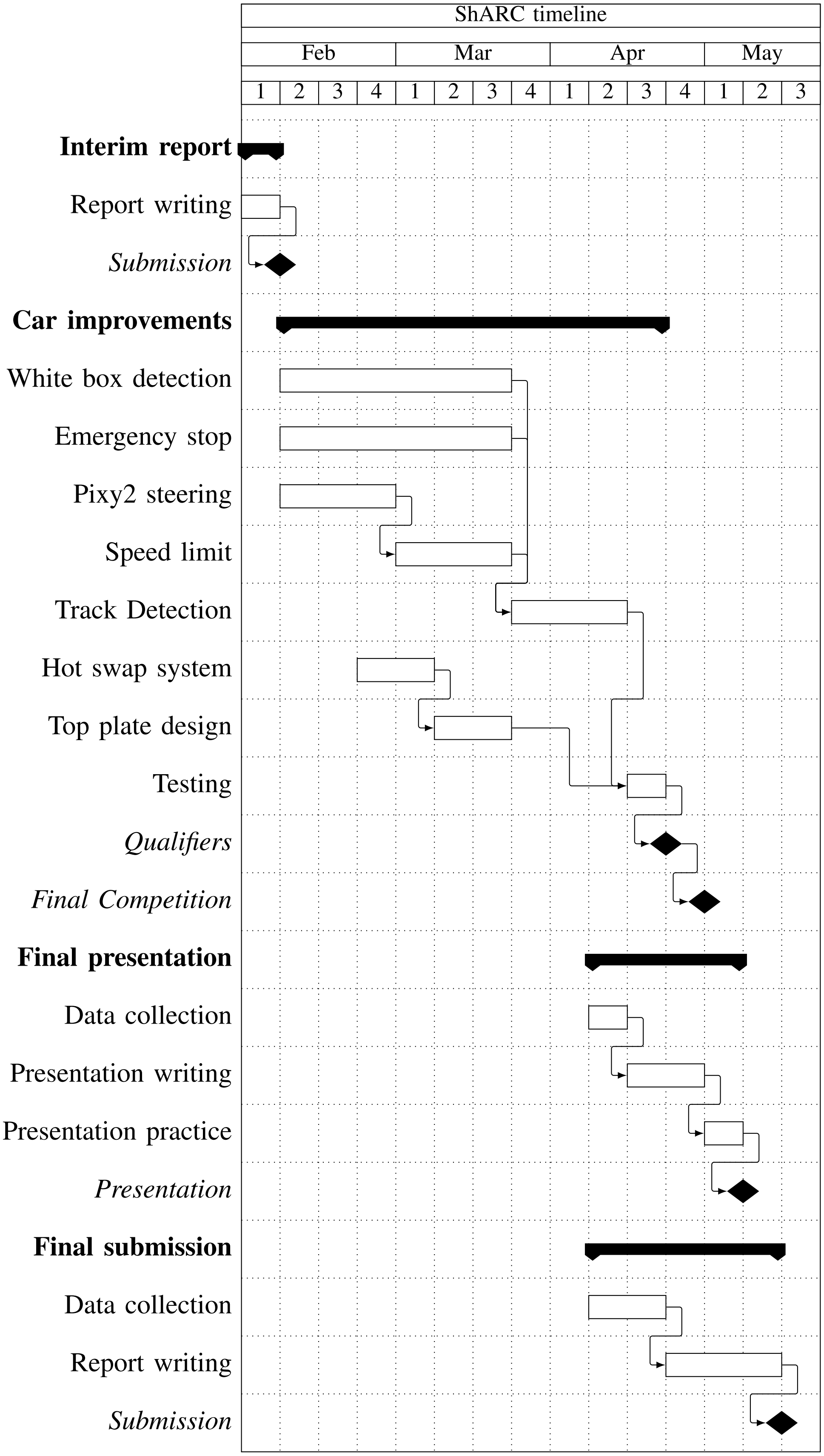

Fig. 1: Gantt chart project plan to meet qualifier and project deadlines

III. PROJECT PROGRESS

A. Steering

1) Component Choice: When determining the servo to purchase for the chosen chassis, reference was taken from the chassis manual, which came with a page detailing an example build – this gave us a combination of components that already worked according to the manufacturer, therefore giving us a minimum specification that was could use to purchase components that matched or exceeded it. From this, the Trackstar TS-920 was chosen, which had an equal rotation speed of 0.07-0.08s/60 degrees, and a greater torque of $13.1kg/cm$ (at $7.4V$), which would give better steering stability. The Trackstar also comes with additional strengths such as its titanium gear for greater durability, ideal for handling the high forces that would be exerted on the servo when cornering.

2) Algorithm: The initial build of the car utilised a proportional controller, which took the average horizonal position of the edges to find the centre point of the track on screen. By comparing this value to the centre of the screen itself, the proportional controller could operate by using the difference as an error input. This method was tested on a straight-line track and found to be functional, however the method was untested on other track components such as bends and chicanes, and more importantly, problems were found with the rate of image acquisition not being fast enough, resulting in delayed responses from the controller. To iterate upon this, a new camera has been purchased which is part of a recommended series of components by NXP for the competition, which comes with its own dedicated processing board and faster interface (I2C versus USB).

3) Mapping: Future iterations of the design hope to implement forms of mapping of the track so as to allow for calculation of the racing line. Until the method of image acquisition is set however, work on how to map the track is not possible, but a program to interpolate between points has been made using a Savinsky-Golay filter which could potentially be used to determine and fill way points on turns to smooth cornering – this program does take time to process depending on the number of points you interpolate and smooth, which limits its effectiveness.

B. Hardware

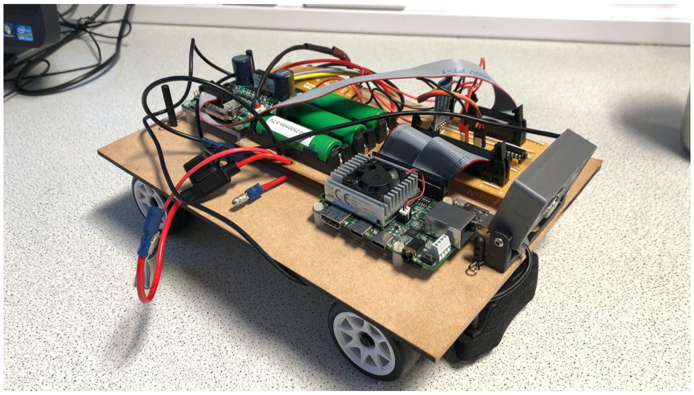

Fig. 2: Photograph of prototype car with electronics board mounted.

1) Chassis: Due to long lead times on the recommended chassis it was decided that an alternative must be chosen. The benefits of a four-wheel-drive drivetrain were extensively discussed in [1]. From this the 3Racing Sakura M4 chassis was chosen. It offered belt-driven 4WD as well as Ackerman style steering, damped spring suspension, brushless motor support and a low maintenance rear gear differential (as opposed to the more complex ball type differential). There is no front differential; while this would reduce steering ability one could be easily added at a later stage. Crucially, the Sakura differed from other chassis by offering an adjustable wheelbase, allowing it to be reduced to $210mm$, $15mm$ shorter than standard in the range and closer to the $180mm$ recommended by NXP. The chassis was assembled within a week and a prototype platform for the electronics was laser-cut from MDF for easy modification. The platform was made to fit over existing support columns for a car body shell.

2) Battery: Lithium chemistry batteries have the highest energy per unit mass/volume of any mainstream chemistry available. Lithium polymer (LiPo) cells are the variant most commonly used in hobby grade model vehicles due to their higher discharge ratings and availability of different shapes. In contrast, Lithium Ion (LiIon) typically has lower discharge current ratings and are available in fewer formfactors but are less volatile and offer longer lifetimes.

LiPo was the first choice for the car as standard sized hobby batteries are readily available and most commonly used, offering very large discharge ratings. However, NXP rules limited LiPo to two cells $(6-8.4V)$ which is below the minimum (typically $8V$) voltage tolerated by more advanced, commercial speed controllers. LiIon cells did not face this limit and so were a good compromise since three could be connected in series to form a $9-12.6V$ battery. ‘High power’ Sony VCT5A LiIon cells in standard 18650 form were chosen. These offer a higher continuous charge/discharge rating than many other power cells, and significantly more than ‘high energy’ cells which offer more capacity at the expense of discharge.

As later calculated, the maximum current draw is around $12A$, well within the $30A$ limit of the VTC5A. Assuming a worst case of continuous acceleration and no regenerative braking and a 60 second lap time, the car can complete 13 laps using the VTC5A’s full $2600mAh$ capacity.

A temporary battery pack mounted to the electronics board has been made while the final cell locations on the chassis are determined. The final aim is to mount the cells to the base-plate of the chassis to lower the centre of mass and free up electronics board space. The cells are held by 18650 holders allowing them to be removed and charged on an external balancing charger, avoiding the need for complex balancing circuitry in the car. A simple low voltage alarm monitors each cell to avoid over discharging, while an inline automotive blade fuse provides over current protection.

3) Motor and Gearing: To set the motor requirements it was necessary to estimate maximum acceleration and speeds expected on the track. The aim was to ensure traction at the tyres was the limiting factor rather than an under-specified motor. Maximum acceleration was derived in [1] and the static coefficient was assumed to be 0.9 based on a dry rubber/asphalt interface [2], giving a traction limited acceleration of $8.8ms^{-2}$.

Looking at the example track, which was roughly similar to previous year’s tracks, the fastest section would occur in the centre of the longest straight (six straight pieces total $4.14m$). At each end are either 90^o or 180^o bends which will limit the car’s entrance and exit speed to the straight based on grip and radius. It was assumed that the car would accelerate at a constant rate limited only by traction up to the fastest speed reached. Assuming a symmetrical track i.e. same bends at both ends and that braking deceleration equalled acceleration, then a simple kinematic equation ($v^2 = u^2 + 2as$) could be used to find maximum velocity at the centre point, once corner speeds are known.

The maximum cornering speed (v_c) based on traction was derived in [1]. This speed depended on the radius of the turn and so the style of driving mattered. The three options based on the standardised track pieces were: following a centre line, following a ‘racing line’ around a 90^o bend and a racing line around a 180^o bend. The centre line calculation was simply finding the mean radius of the inner and outer track radii. This was found to be $0.445m$, giving a v_c of $1.98ms^{-1}$.

The racing line radius was calculated by finding the largest radius arc which could fit inside the track. This can be simply derived geometrically [3]. The race radius (r_r) depends on the inner track radius (r_i), outer track radius (r_o) and angle of the turn (θ):

$$r_r = \frac{r_i \cos(\frac{\pi}{2}) - r_o}{\cos(\frac{\pi}{2}) - 1} \tag{1}$$

For the 90^o bend, r_r was found to be $2.05m$, giving a v_c of $4.25ms^{-1}$. For the 180^o bend, r_r was found to be $0.72m$, giving a v_c of $2.52ms^{-1}$.

For the centre line, it can be assumed that maximum acceleration begins as soon as the straight track starts. However, the racing lines extend into the straight sections of track and so continue to limit speed after the car has left the corner track pieces. Therefore, the displacement (s) in the kinematic equation for the two types of bends needs to be reduced by r_r minus either r_i or r_o depending on the type of bend as shown in Fig. 3 and Fig. 4. From these assumptions the maximum speed for the straight between two 90^o bends was $5.58ms^{-1}$ and $5.75ms^{-1}$ for two 180^o bends. For the simpler centre line case the maximum speed was $6.36ms^{-1}$. The highest and hence worst case speed was for the centre line and so this speed will be used for motor calculations.

Hobby motors are typically specified by their K_v (RPM/V) rating and by their current rating. K_v is inversely proportional to the back EMF constant (K_e) but is measured line-line rather than line-neutral. For trapezoidal motors such as these, the torque constant (K_T) is equal to $2K_e$. Assuming they use 6-step commutation, then the line-line voltage is twice line-neutral, meaning K_T equals $K_{e(line-line)}$ which can be found by inverting K_v and converting to rad/s:

$$K_T = 2K_e = K_{e(line-line)} \tag{2}$$

$$K_T = \frac{60}{2\pi}\frac{1}{K_V} = \frac{9.54}{K_v} \tag{3}$$

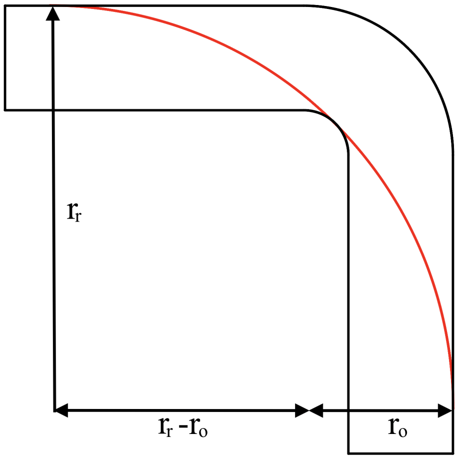

Fig. 3: Racing line for 180^o bend.

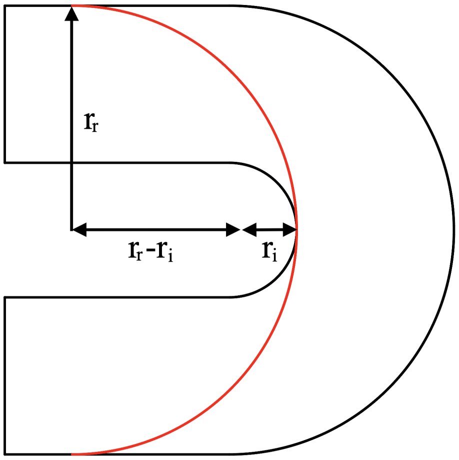

Fig. 4: Racing line for 90^o bend.

Winding resistance is not always given however, researching other motors in other ranges produced by the same manufacturer revealed that stall current is typically 10-20 times greater than rated current. For a worst case, rated current was multiplied by 10 to find stall current. Stall and rated torque are therefore given as below, along with no-load speed in rad/s. V_m is the maximum motor line-line voltage ($8.4V$) based on the maximum voltage of a 2 cell LiPo which they are rated for:

$$\tau = \frac{9.54I}{K_v} \tag{4}$$

$$\omega_0 = \frac{2\pi}{60} K_v V_m \tag{5}$$

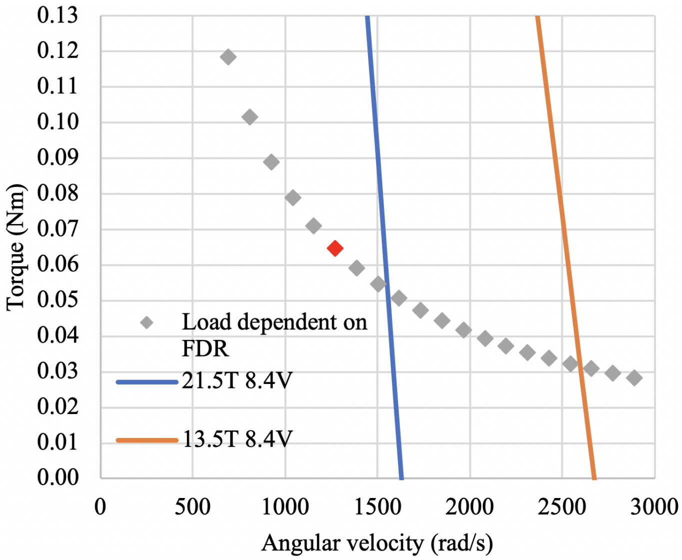

Fig. 5: Motor characteristics and load line. Torque axis terminates at rated torque, assumed to be 10% of stall torque. Red load point represents an FDR of 5.5, roughly that of the existing 80-tooth spur and 28-tooth pinion. The FDR increases by 0.5 from left to right.

Gearing is also a variable which must be determined. The load the car presents to the motor after gearing was plotted from the required speed and acceleration by varying the final drive ratio (FDR – total gearing including drivetrain ratio of 2). The spur and pinon gears are specified by their tooth count (T_s and T_p) and the spur included with the chassis was 80-tooth. A 28-tooth pinion was ordered as this was recommended by the manufacturer as a starting point.

$$FDR = 2\frac{T_s}{T_p} \tag{6}$$

The angular velocity (ω) and torque (τ) at the motor shaft are based on the maximum speed reached (v), wheel diameter (D, $55mm$), static coefficient of friction (μs, 0.9), car mass (m, $1.47kg$) and acceleration due to gravity (g).

$$\tau = \frac{\mu_s F_n \frac{D}{2}}{FDR} = \frac{\mu_s m g D}{2FDR} \tag{7}$$

Fig. 5 was produced for the most likely two motors, using the recommended motor as a starting point. The motor lines were plotted using the no load speed and stall torque calculated above. Hobby motors are typically referred to by their turn count. The $13.5T$ motor is rated at $8.4V$, $36A$, $3040RPM/V$. The $21.5T$ motor is rated at $8.4V$, $22A$, $1855RPM/V$.

From this the $21.5T$ motor was chosen and the existing FDR kept. The lower current draw of this motor would reduce load on the system, increasing the battery and controller options available. The controller will also be able to use a larger range of its output in driving the motor, giving better precision. The motor current draw at maximum acceleration was calculated from the torque equation above and found to be $12A$.

4) Motor Controller: Both hobby and more industrial controllers were researched for use in the car. Hobby controllers were cheap, had high current capabilities, offered braking and basic 'traction control'. However they did not offer any closed loop speed control and braking was achieved by requesting levels of 'reverse' during forward motion. They were most likely using sign magnitude drive, using the body diodes of the MOSFETs to achieve discontinuous current flow when forward throttle was applied but less than the current speed (i.e. $BEMF > V_m$). This means the motor can both freewheel and brake, the exact response of which is unknown, and so building an external closed loop controller could be very complicated.

Looking solely at closed-loop controllers, all offer closed loop speed and either torque or current limiting for advanced traction control. VESC and ODrive are both open source controllers, however, due to delivery/import costs, for the price of these controllers proprietary commercial alternatives could be bought which were more likely to 'just work' and offer simpler programming interfaces. Two controllers recommended by a previous race winner and researcher were Maxon 50/8 and Roboteq SBL1360. The SBL offered a more advanced RS232 interface and had higher current ratings, as well as being available from a UK seller and so it was chosen.

The SBL has so far been interfaced with both a FRDM-k64f microcontroller and the Coral Dev Board. Speed, currents etc. can be read and the computers can issue speed commands to the controller. The SBL has been set up to offer integral only speed control, with voltage limiting to 8.4V to avoid damaging the motor and acceleration limiting to avoid wheelspin. A watchdog timeout has also been enabled, so that the car safely stops in case of a bug with the team's code.

5) Miscellaneous electronics: A bidirectional level shifter was made so that the microcontroller could control the high voltage servo. The circuit was made with two $10k$ pullup resistors and a 2N7000G NMOS. While testing with a signal generator resulted in servo jitter, using the hardware PWM generators on the microcontrollers produced sufficiently smooth movement. To interface between the polar RS232 code of the SBL and unipolar UART code of the two microcontrollers, a MAX3232 level shifter chip was bought. A simple supporting circuit was constructed for it based on the one given in the datasheet. A HC-05 serial Bluetooth module was used to establish simple wireless control of the Coral Dev Board. The module was connected to the board's serial port so that it was essentially transparent and could be connected to as if over a wired connection. The above three systems were all built on one matrix board to save space. The board was connected directly to the Coral Dev Board's I/O pins using a 40-pin IDC connector and ribbon cable.

The electronics are all powered by an off the shelf switching regulator, able to provide both $7.4V$ for the servo and $5V$ for everything else.

C. Computer vision

Currently the car detects where to head next based purely from the input from the camera. The algorithm for this is very basic currently where the image captured is processed by OpenCV by being split into left and right sections, each of the two sections is looking for a black lane by converting the image to grayscale, gaussian blurring the image and then linearly thresholding to black or white. The black regions are then averaged in the x axis giving an average future point, this done on the left and right lanes allows the average centre to be produced by averaging these two points giving a single desired point in the future that proportionally controls the angle of the wheels. This can be seen in operation in Fig. 6. This system however is prone to issues surrounding bad lighting as shiny black lines can appear white in bright lights, a method to improve this would be to instead use gaussian thresholding increasing true positive rate but unfortunately will also increase the false positive rate leading to a noisier and more incorrect prediction. This method was implemented over the previous Hough transform and line detection due to issues detecting broken lines from bad lighting and the impossibility of detecting corners using this method. Other methods explored suffered the same fate of being unable to deal with broken curves or imperfect data.

Currently the data feeding into the computer vision is from a desktop screen mount style camera, this camera is accessed over the USB interface and unfortunately has a lag from capture to processing,

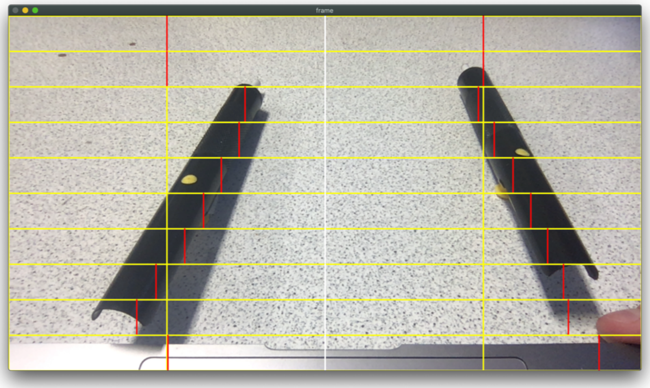

Fig. 6: Black averaging algorithm currently used

this means that the car is constantly hundreds of milliseconds behind the actual position, this likely occurs from a mix of on camera processing and USB/Operating system interface lag and is clearly unacceptable for a high speed self-driving car but works at low speeds. To get around this a new camera has been purchased that is designed for this application and can be interfaced over I2C without the need of an operating system, this is yet to be implemented.

1) Coral Dev Board : The main processing board, coral dev board, is currently being used to control everything whilst also running the computer vision software, this shouldn't be an issue but means a lot of pressure is on the board for computing everything in real time, this is achieved through a barebones Linux operating system called Mendel. This allows multiprogramming controlling all aspects of the car concurrently. This however means constant IO calls force system calls to complete operations which is significantly slower than in user space. These blocking IO calls through system calls leads to a noticeable drop in processing speed and can easily lead to CPU idling. This is also not helped by the supported packages for accessing these system calls and for OpenCV is Python 3 which is an interpreted language once again not famed for its speed and use in real time systems. As every aspect in this system is using interpreted system calls moving away from this operating system architecture and instead embedded compiled instructions may be wise and can be operated via lightweight slave master processor architectures.

REFERENCES

[1] A. Huff, D. Chung, and H. Sams, *Project Initialisation Report NXP Cup 2020 EMEA Car - 4th Year Group Project*, 2019.

[2] "Friction and friction coefficients." [Online]. Available: https://www.engineeringtoolbox.com/friction-coefficients-d_778.html

[3] "Car racing: How to calculate the radius of the racing line through a turn of varying length." [Online]. Available: https://math.stackexchange.com/q/2992667