CRANFIELD UNIVERSITY

**Software Engineering for Technical Computing MSc**

# VvortexX

Investigation of Vortex Behaviour For Practical Application

***Group Members:*** *David Wang - S369748 Daniel Smythe - S374685 Vetrichelvan Pugazendi - S351400 Pierre Cisse - S368793 Imad lL Islam - S371584 Itohanoghosa Eregie - S356017 Tae Yong Kim - S332600*

***Supervised by:*** *Irene Moulitsas - i.moulitsas@cranfield.ac.uk, Seemal Asif - s.asif@cranfield.ac.uk, Tom Teschner - tom.teschner@cranfield.ac.uk, Jun Li - Jun.Li@cranfield.ac.uk*

## Abstract

Understanding and analyzing fluid flows is one of the most challenging areas of research. A strong grasp of these concepts can lead to better aerodynamic designs of products. The aim of this project is to build technologies that are able to find vortices and their cores that occur in the wake region, formed behind golf balls. In this project, a complete application to detect vortices and its cores is developed. The team consists of Machine learning engineers, Computer Vision Engineers, Computational Fluid Dynamic Engineers and Software Engineers.

The application consists of three parts:

The Simulation - here the CFD engineers simulates fluid flow over a golf ball and provides images which contain vortices,

The Front End - The frontend through which the users can interact with the object detection model is built by the software engineers.

The Back End - which consists of the YOLOv5 model with a custom core detection algorithm is built by the machine learning and computer vision engineers. This application successfully detects vortices and their cores with a mean average precision of 92%. In addition we produce an estimate to the vortex core for each vortex, which displayed more accurate results than the naive approach.

# Contents

# List of Figures

# List of Tables

# 1 Introduction

## 1.1 Background

Vortices are an extremely important part of any application or industry which need to consider aerodynamics in the design of their product. A vortex can be simply defined as a region in a fluid by which the flow in that region resolves around some sort of axis. This gives us the circular characteristic commonly associated with vortices in media such as in fluids such as whirlpools or tornadoes.

When we have regions where many vortices form, they tend to interact heavily with each other and due to this, drag due to the friction with the fluid particles tends to increase. It is therefore important for someone working with aerodynamics to be able to identify vortices during their tests and simulations.

There are many ways to identify vortices using the numerical information about pressure and other such values within a simulation. However, there have not been many applications which aim to identify vortices though modern object detection algorithms. From simulations of these fluid systems we are able to produce images displaying a visual representation of the flow in a system. In addition we can produce this sort of imagery in a real-life system by emitting smoke particles and filming their movement.

Object detection algorithms such as YOLO are already shown to achieve very accurate results. By using object detection models to find vortices in an image we are additionally using the knowledge and understandings developed by an entirely different area of research to a different problem, which might provide new insights and understandings to both topics of research. The aim here is to potentially develop alternative methods to detect vortices in images and simulations and to enrich the understand of vortices by applying methods from entirely different fields to a very similar situation. In addition to potentially replacing more classic methods of vortex detection, object detection models could potentially be used to enhance preexisting methods in unexpected ways.

In addition, sometimes we might not have access to the data necessary, or the data might be difficult to obtain, when detecting vortices using classical methods. By specifically aiming to detect objects within images there is potential to bring vortex detection to a much wider field in which the data is inaccessible, but images are easy to gather in the modern era. By targeting this specific area there is potential to create applications that target a wider audience than with the data-dependent methods.

## 1.2 Market Analysis

Based on our findings, there are few vortex detection software in the market that is cloud based with micro-services architecture.A business cannot exist without customers therefore, it is important to define the customer and target market in order to properly design the product, define marketing strategy to target the customers, pricing and fix promotions. This analysis describes what our company will market (the product), how and when we will market the product(packaging and distribution), our target market(customers) and the price.

### 1.2.1 Our Market

As a software development company, we have identified our target market as companies that specializes in the design aerodynamic objects. We have Identified two major markets:

- Aerodynamic market for automotive
- Golf ball production companies

The aerodynamic market for automotive is projected to grow at a compound annual growth rate of 4.77 percent from 2018 to 2025 according to https://www.marketsandmarkets.com/ and for golf ball USD 1.38 billion by 2025, according to a study by Grand View Research Inc. It is projected to expand at a compound annual growth rate of 2.7 percent over the forecast period according to https://www.grandviewresearch.com/. The market is growing.

### 1.2.2 The Product

Our company offers a Vortex Detection system that leverages on machine learning and computer vision to detect vortices and vortex cores. Our product is highly competitive in the market because it is cloud based. The customers are buying a product that is very robust, agile and easily accessible from anywhere because it is cloud based. Customers will prefer our product to the ones existing because it is cloud based, easily scaled and can be deployed on premises. The product was developed using agile methodology which gives room for further development and improvement.

### 1.2.3 Product Distribution

The product is a cloud based software and will be available over the internet. Customers will have the ability to run demos before purchasing it.

### 1.2.4 Pricing

The pricing model for our product is competitive pricing. This will enable our company match the current price of the few vortex detection software in the market.

### 1.2.5 Promotions

Our strategy for promotions is to offer proof of concept to potential customers. By advertising our promotions on high traffic websites and targeting customers in aerodynamics, we will gain more market grounds. Currently, advertising via social media is the most effective way to reach customers and that is the strategy we will employ.

### 1.2.6 Marketing Strategy

Our marketing strategy is to focus on satisfying our customers while making the product more affordable than the existing products in the market.

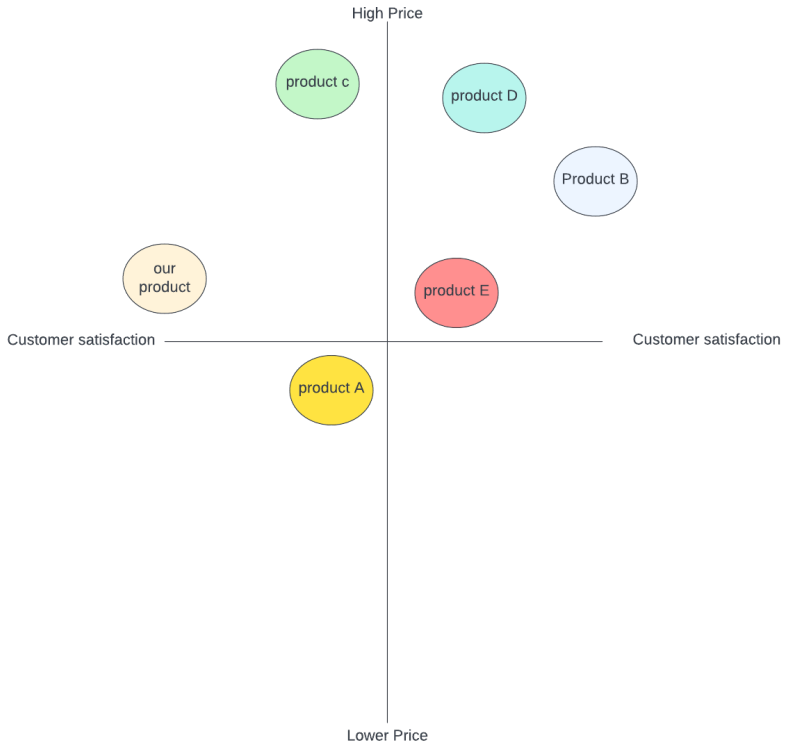



Figure 1: Competitive Landscape

### 1.2.7 Aims and Objectives

From the stated Market Analysis we can outline the following objectives for this project:

- Investigate the variation in vortices produced when using different variations of a Golf Ball Simulation.
- Develop a machine learning model which is able to detect the locations of vortices within an image.
- investigate the use of classical Computer Vision techniques in determining Vortices and the locations of the Vortex Cores.
- Combine the development explored by in the other objectives to produce an application that applies the method explored for those objectives. The aim of this is to produce a competitive product satisfying the customer needs and providing a unique and accurate approach to the topic.

# 2 Literature Review

## 2.1 Overview of Literature review and theory

### 2.1.1 Literature Review

From the detection of vortices to their meticulous study for investigating the fluidic implications that vortices suggest, scientists and researchers have passionate interests in the scientific investigation of what revolves around vortices.

This observation is even more relevant when the multiple definitions of what a vortex is, converge towards a certain conclusion: a vortex is a common phenomenon in fluid mechanics which impacts the environment of its appearance.

Inextricably linked to lift and drag but also to combustion and propulsion (chemical physics), the behavior of vortices in a medium reveals the dynamics of this medium. Thus, the investigation of the behavior vortex is of real necessity when it comes to possible practical applications.

The purpose of this literature review is to address the scientific landscape of the investigation of vortex behavior for practical applications. The sole aim is to build a sustained understanding of the existing and to give a global vision of the gap with the narrower subject that supports this document: Investigation of vortex behavior for designing aerodynamic shape (or hydrodynamic shape) - case of Golf Ball. The review will address certain relevant articles to support the direction taken in the realization of the project already mentioned. This part of the literature review is limited to the computational fluid dynamics horizon of the project with some appeals to the AI side of the project which is inseparable from it.

A first observation can be made before getting into the hard part of the articles: the absence of a solution in exact compliance with the specifications of the project. No document detailing a fluidic study of an aerodynamic (or hydrodynamic) object empowered by the contribution of artificial intelligence for the optimization of the design of a more efficient shape.

**Theoretical perspective**
Vortex potential method and Contour sum method from Murphy and Dainty, 2012 respectively were first proposed by LeBigot and Wild in the late 90's and recommended by Fried in 1992 as explained in the article by Kevin Murphy and Chris Dainty. The Vortex potential method involves rotating all of the phase gradient values calculated from the wavefront sensor by 90 degrees and then forming a potential plot of the resulting field using least-squares reconstruction, but The Contour sum method is essentially a closed line integral over the phase function's gradient: two being computational methods which are then automated. Applied to detecting optical vortices from Shack-Hartmann wavefront sensor (SHWFS) data, the results show that the vortex potential method is robust for single vortex detection; in the field, the percentage of correct detections is frequently greater than 90% and The contour sum method performs significantly worse than the vortex potential method under all conditions, poorly at high sampling levels. Interesting results but still far from opening the doors of detection for aerodynamics: nevertheless, the means of detection can be interesting if they are applied to the intrinsic subject.

Because it is based on the velocity field topology and does not use velocity derivatives

that are sensitive to spurious vectors, the $\Gamma_2$-criterion method is a robust and reliable alternative as investigated by Fabrizio De Gregorio and Antonio Visingardi De Gregorio et al., 2019 (to identify and characterise vortical structures in a flow field measured with a traditional two-component PIV measurement system). The 2-criterion performs admirably in terms of robustness and reliability on real PIV data, especially in the presence of strong laser reflections in the measurement region: relying on spatial derivatives of the velocity field. Some of its serious (but less successful) competitors are The Q-criterion (one of Hunt et al.'s popular vortex identification methods) (1988). Q is defined as the residual of the vorticity tensor norm squared subtracted from the strain-rate tensor norm squared) and The $\Delta$-criterion (Chong et al. (1990) define a vortex core as the region where $\nabla v$ has complex eigenvalues, so the criterion is based on the discriminant of the characteristic equation for the velocity gradient tensor $\nabla v$ study).

**Practical perspective**

To go into the "practical", "applicative" aspect of the study (investigation) of vortices, for narrowing the review: let us tackle the research (articles) that go beyond "calculative" detection but go as far as the application of the extracted results to the understanding of the behavior of vortices.

A high-fidelity numerical simulation of expiratory biosol transport during normal breathing under indoor, stagnant air conditions with and without a facial mask investigates mask efficacy to suppress the spread of saliva particles, which is the basis for current social distancing recommendations by Khosronejad et al., 2021.

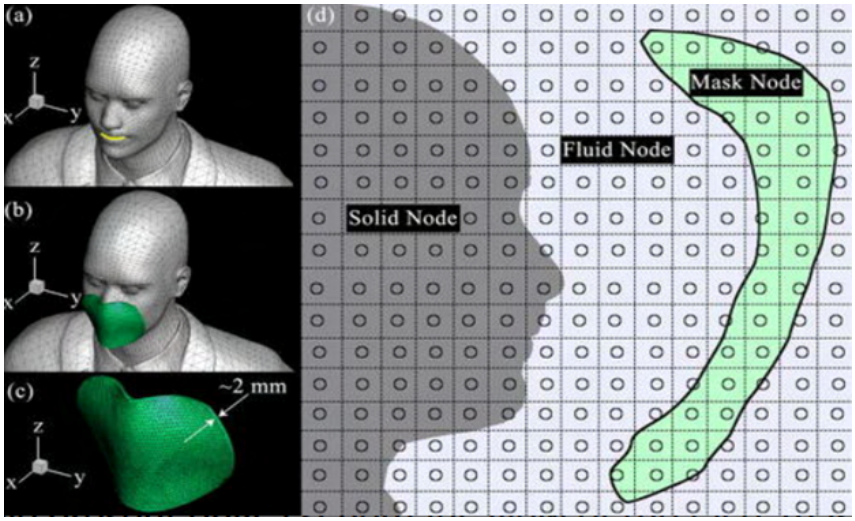



Figure 2: Biosol transport during normal breathing

The simulations account for the effect of human anatomy as well as a range of saliva particulate sizes ranging from 0.1 to 10 m, as well as their evaporation. The simulations reveal the vorticity dynamics of human breathing and demonstrate that without a facial mask, saliva particulates can travel more than 2.2 m away from the person. This study proves that the investigation of vortices is not a scholar's whim but of scientific utility (sometimes social).

The previous article echoes the study of "Efficient collective swimming by harnessing vortices through deep reinforcement learning" by Verma et al., 2018 which aims (via the study of vortices) to facilitate the design of "robotic swarms" (which could revolutionize medical technology, for example). The understanding of the physical mechanisms used by active swimmers in nature (nektons) is also a long-standing interest in the scientific

world. Surprisingly, they discover that swimming behind a leader is not always associated with increased energy levels in the follower.

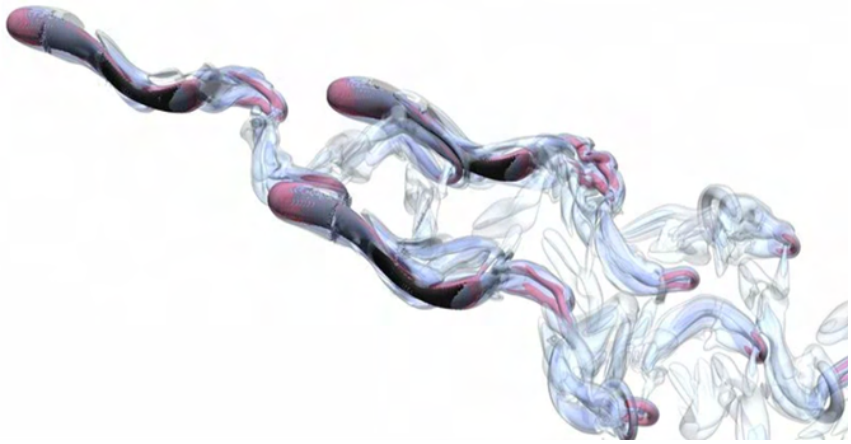


Figure 3: Efficient collective swimming

In turn, they show that fish can improve their sustained propulsive efficiency by strategically positioning themselves in the wakes of other swimmers and intercepting their wake vortices.

Vortex identification and visualisation are critical for understanding the underlying physical mechanism of the flow field and have recently received a lot of attention: it is, therefore, logical to see experts in AI take an interest in it (which we will do a posteriori). Local vortex identification methods can provide results quickly, but they require the selection of an appropriate criterion and threshold, which leads to poor robustness. Global vortex identification methods have the potential to produce reliable results, but they require significant user input and are computationally intractable for large-scale data sets. To address the problems described above, a novel vortex identification method based on convolutional neural networks (CNN) is proposed by Liang et al., 2018 to combine the benefits of both local and global vortex identification methods and achieve higher precision.

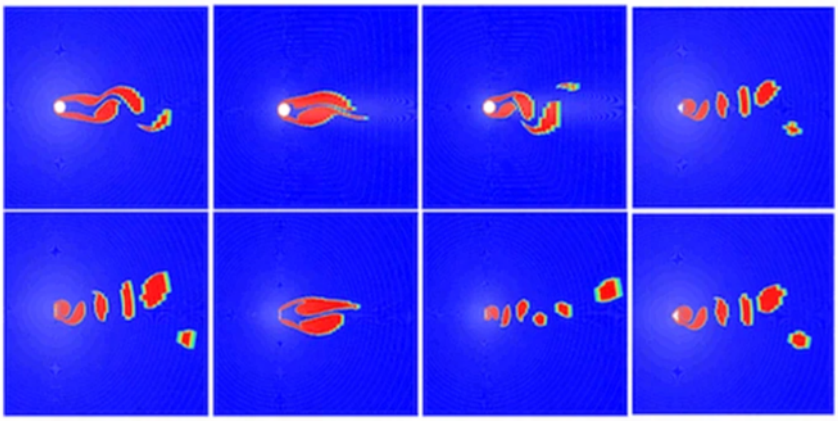


Figure 4: Efficient collective swimming

The proposed method achieves higher precision and recall than local methods and achieves a speedup of more than 6 times over the global and objective methods. In comparison to traditional machine learning algorithms and MLP, the approach outperforms them and achieves good visual effects. The last article to be analyzed rather judiciously captures the essence of the project suggested by this literature review: “Aerodynamic shape effects

of tall building for vortex-induced vibration" by Hayashida and Iwasa, 1990. From these results, there is a dynamic response behaviour of a super high-rise building under strong wind due to vortex shedding, as well as an aerodynamic damping effect of vibration by changing the cross-section of the building. This ends up convincing of the importance of the study undertaken but also of the gaping absence of the appropriate tool (the one that will combine vortex study and the objective of optimizing the design of object shapes in a scientific marriage between CFD and AI).

There is no shortage of "mathematical" and then "computing" tools for the study of vortices, attempts to understand this phenomenon with the aim of optimizing the design of aero / hydrodynamic objects are numerous but there is not a single combination, judicious, of the two which will facilitate the work of design (by automation). Hence the current project.

### 2.1.2 Theory

For the Computational Fluid Dynamics part of this project, several simulations are carried out (they will be detailed in the next sections). To understand the choices of models and methods inherent in simulations, one discusses in this part some theoretical notions behind the Fluent software (in terms of model).

*Viscous Model:*

**K-epsilon Realizable:**

ANSYS FLUENT employs a variety of closure models to describe the effects of turbulent fluctuations in velocities and scalar quantities in a single phase. In comparison to single-phase flows, the number of terms to model in the momentum equations in multiphase flows is large, making turbulence modelling in multiphase simulations extremely complex. Within the context of the k-epsilon models, ANSYS FLUENT provides three methods for modelling turbulence in multiphase flows. Furthermore, within the context of the Reynolds stress models, ANSYS FLUENT provides two turbulence options (RSM). There are three k-epsilon turbulence models to choose from: mixture turbulence (the default), dispersed turbulence, and turbulence model for each phase.

The realisable k-epsilon model is a recent development that distinguishes from the standard k-epsilon model in two significant ways: A new transport equation for the dissipation rate, epsilon, has been derived from an exact equation for the transport of the mean-square vorticity fluctuation in the realisable k-epsilon model.

The term "realisable" shows that the model fulfils certain numerical constraints on the Reynolds stresses, which are consistent with turbulent flow physics. The standard k-epsilon model and the RNG k-epsilon model are both impractical. The realisable k-epsilon model has the immediate advantage of more accurately predicting the spreading rate of both planar and round jets. It is also likely to outperform in flows involving rotation, boundary layers with strong adverse pressure gradients, separation, and recirculation.

In the realisable k-epsilon model, the modelled transport equations for k and epsilon are:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_j}(\rho k u_j) = \frac{\partial}{\partial x_j}\left(\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right) + G_k + G_b - \rho_\epsilon - Y_M + S_k$$

And

$$\frac{\partial}{\partial t}(\rho\epsilon) + \frac{\partial}{\partial x_j}(\rho\epsilon u_j) = \frac{\partial}{\partial x_j}\left(\left(\mu + \frac{\mu_t}{\sigma_\epsilon}\right)\frac{\partial \epsilon}{\partial x_j}\right) + \rho C_1 S_\epsilon - \rho C_2 \frac{\epsilon^2}{k + \sqrt{v\epsilon}} + C_{1\epsilon}\frac{\epsilon}{k} C_{3\epsilon} G_b + S_\epsilon$$

Where:

$$C_1 = \max\left(0.43, \frac{\eta}{\eta + 5}\right), \eta = S\frac{k}{\epsilon}, S = \sqrt{2S_{ij}S_{ij}}$$

The eddy viscosity is calculated from:

$$\mu_t = \rho C_\mu \frac{k^2}{\epsilon}, C_{1\epsilon} = 1.44, C_2 = 1.9, \sigma_k = 1.0, \sigma_\epsilon = 1.2$$

**K-Omega SST:**

One of the most widely used models for capturing the influence of turbulent flow conditions is the k-omega turbulence model. It is a member of the Reynolds-averaged Navier-Stokes (RANS) relatives of turbulence models, which models all turbulence effects.

It is a model with two equations. That is, furthermore, the conservation equations, it aims to solve two transport equations (PDEs) that account for history effects such as convection and turbulent energy diffusion. Turbulent kinetic energy (k), which specifies the energy in turbulence, and specific turbulent dissipation rate (), which measures the rate of dissipation per unit of turbulent kinetic energy, are the two transported variables. is also referred to as the turbulence scale. The standard k model has a low Re. SST is an abbreviation for shear stress transport. In the free-stream, the SST formulation switches to a k behaviour, which avoids the k issue of becoming responsive to the inlet free-stream turbulence characteristics.

The k SST model predicts flow separation better than most RANS models and accounts for its good behaviour in adverse pressure gradients. It can account for the principal shear stress transport in adverse pressure gradient boundary layers. Because of its high accuracy-to-expense ratio, it is the most popular model in the industry.

On the downside, the SST model generates a lot of turbulence in areas with a lot of normal strain, such as stagnation zones and areas with a lot of acceleration.

The following formula can be used to calculate the specific turbulent dissipation rate:

$$\omega = C_\mu^{\frac{3}{4}} \frac{k^{\frac{1}{2}}}{l}$$

And the turbulent viscosity is calculated as follows:

$$v_t = \frac{k}{\omega}$$

**Transition SST:**

In terms of momentum-thickness Reynolds number, the transition SST model relies on the conjugation of the SST k-omega transport equations with two other transport equations, one for the intermittency and one for the transition onset criteria. An ANSYS specialised empirical relation (Langtry and Menter) has been devised to cover standard bypass transition flows as well as flows with low free-stream turbulence.

**Transition K-Kl Omega:**

This model has the following characteristics: incompressible only, three-equation model, low Reynolds number, and transition modelling.

The equation for specific dissipation rate:

$$\frac{D}{Dt}(\omega) = \nabla.(D_\omega \nabla_\omega) + C_{\omega 1} P_{kt} \frac{\omega}{k_t} - \left(1.0 - \frac{C_{\omega R}}{f_\omega}\right) k_l (R_{bp} + R_{nat}) \frac{\omega}{k_t} - C_{\omega 2} f_\omega^2 \omega^2 + C_{\omega 3} f_\omega \alpha_t f_\omega^2 \frac{k_t^{0.5}}{y^3}$$

The equation for laminar kinetic energy:

$$\frac{D}{Dt}(k_l) = \nabla.(v \nabla k_l) + P_{kl} - R_{bp} + R_{nat} + D_l$$

The equation for turbulent kinetic energy:

$$\frac{D}{Dt}(k_t) = \nabla.(D_k \nabla k_t) + P_{kt} + (R_{bp} + R_{nat}) k_l - \omega + D_t$$

**Reynolds Stress:**

The Reynolds Stress terms can be used to derive a transport equation.

- The model is more complex because there are more equations to solve.
- There are additional unknowns, which necessitate the use of a model.
- They do, however, take an important step by allowing the Reynolds.

| Model | Description |
|---|---|
| Spalart – Allmaras | A single transport equation model solving directly for a modified turbulent viscosity. Designed specifically for aerospace applications involving wall-bounded flows on a fine near-wall mesh. FLUENT's implementation allows the use of coarser meshes. Option to include strain rate in k production term improves predictions of vortical flows. |
| Standard k–ε | The baseline two-transport-equation model solving for k and ε. This is the default k–ε model. Coefficients are empirically derived; valid for fully turbulent flows only. Options to account for viscous heating, buoyancy, and compressibility are shared with other k–ε models. |
| RNG k–ε | A variant of the standard k–ε model. Equations and coefficients are analytically derived. Significant changes in the ε equation improves the ability to model highly strained flows. Additional options aid in predicting swirling and low Reynolds number flows. |
| Realizable k–ε | A variant of the standard k–ε model. Its "realizability" stems from changes that allow certain mathematical constraints to be obeyed which ultimately improves the performance of this model. |
| Standard k–ω | A two-transport-equation model solving for k and ω, the specific dissipation rate (ε / k) based on Wilcox (1998). This is the default k–ω model. Demonstrates superior performance for wall-bounded and low Reynolds number flows. Shows potential for predicting transition. Options account for transitional, free shear, and compressible flows. |
| SST k–ω | A variant of the standard k–ω model. Combines the original Wilcox model for use near walls and the standard k–ε model away from walls using a blending function. Also limits turbulent viscosity to guarantee that $\tau_T \sim k$. The transition and shearing options are borrowed from standard k–ω. No option to include compressibility. |
| Reynolds Stress | Reynolds stresses are solved directly using transport equations, avoiding isotropic viscosity assumption of other models. Use for highly swirling flows. Quadratic pressure-strain option improves performance for many basic shear flows. |

Figure 5: Summary comparison ()

## 2.2 Aerodynamics

Aerodynamics is the branch of physics that deals with the motion of air around a model. Any model that flows through air has aerodynamic forces acting on them. There are four Aerodynamic forces acting on a body that moves through air, namely:

- Lift
- Drag
- Thrust
- Weight

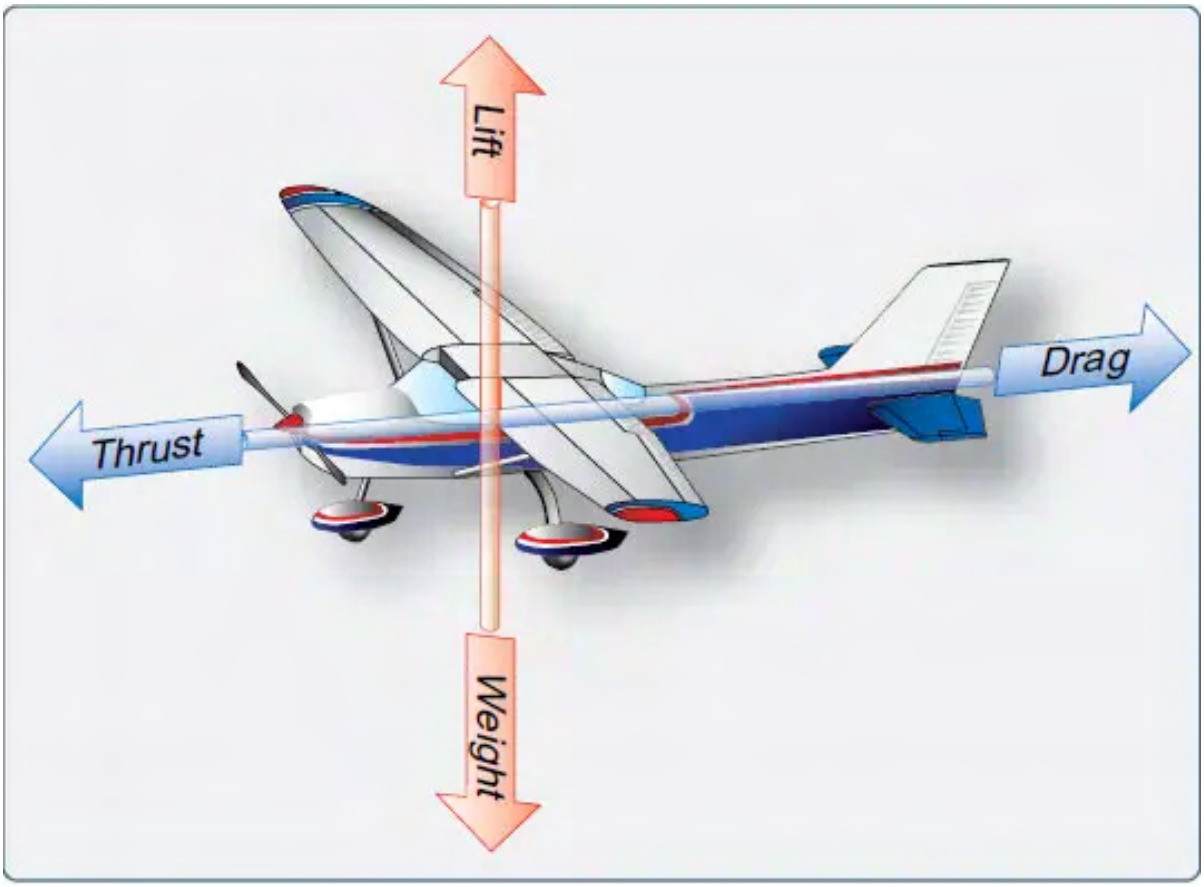



Figure 6: Aerodynamic Forces acting on an Aircraft https://www.flight-mechanic.com

Lift is the upwards force. Anything that flies should generate enough lift to fly. When the air flows around the wing of an aircraft. The flow on the upper surface of the wing has higher velocity than the flow on the lower surface, resulting in lower pressure on the upper surface and higher pressure on the lower surface of the wing, generating lift.

Lift Coefficient is calculated using equation

$$C_L = \frac{L}{\frac{1}{2} \times \rho \times v^2 \times A} \tag{1}$$

Drag force slows down the object. The shape of the object influences the drag force exerted on the body.

Drag Coefficient is calculated using equation

$$C_D = \frac{D}{\frac{1}{2} \times \rho \times v^2 \times A} \tag{2}$$

Thrust acts opposite to Drag force, thrust is a forward force. An aircraft can move forward when the thrust is greater than the drag force. In a propeller driven aircraft, the propeller produces enough thrust for the aircraft to move. Jet Engines produce thrust in a Commercial Aircraft. A glider aircraft does not produce any thrust, it glides through air until drag slows down the aircraft. Weight is the gravitational force acting on the body.

### 2.2.1 Turbulence

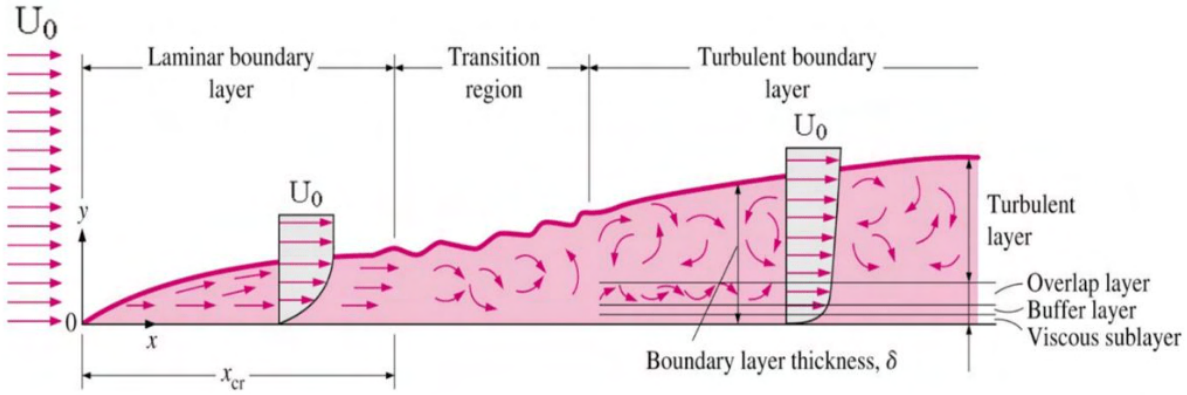



Figure 7: Boundary Layer formation on a flat plate https://www.nuclear-power.com

As the flow flows over the surface, Boundary Layer if formed, when the flow reaches the transition point, the boundary layer transitions to Turbulent Boundary Layer from Laminar Boundary Layer, as the flow tends to separate from the surface. Turbulent flows are chaotic and capricious eddies of air.

Wake Vortex Turbulence is defined as the disturbance in air that is generated behind the body. Wake vortex turbulence can be dangerous depending on the strength of the wake region. Wake from bigger aircrafts can cause smaller aircrafts to stall and eventually crash. Although they are dangerous the wake vortex are short lived. The wake vortex is caused by the pressure differential of the flow, the pressure differential triggers the flow to roll aft the body, resulting in chaotic reverse flow trailing downstream the body.Donahue, 2019

The Strength of the vortex is defined by the weight, speed and shape of the body the air flows over. In Aircrafts, the vortex characterstics are changed by the use of high-lift devices such as flaps, slats etc., However, the vortex strength increases proportionally to weight of the body. Vortices tend to last longer under stable air conditions, when the wind speed is low. when the wind speed is low at higher altitudes the vortices tend to last longer because of the low air density. Vortices descend upon formation until they decay.Donahue, 2019

In this test case, the wake vortex turbulence of Golf Ball is studied and compared with smooth ball. The Same aerodynamics principles apply for all cases. Hence, it is important to study the behaviour of vortex to avoid accidents or incidents. The Wake Vortex Turbulence is the strongest when a body follows to the same path, the body can either be an aircraft or a ball.

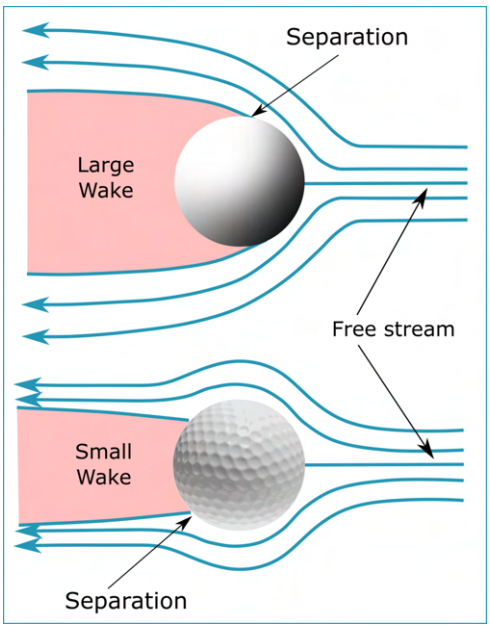



Figure 8: Flow Separation Over Smooth and Golf Ball
https://www.designreview.byu.edu

## 2.3 Object Detection

Small object detection is an ongoing field of research that can have a huge impact on the field of computer vision. To successfully detect small objects, according to Liu et al., 2021, a few challenges need to be overcome.

Challenge 1: Individual feature layers do not contain sufficient information for small object detection as stated in Liu et al., 2021.

As convolutional networks abstract an object to its low level features, small images already having very small features get lost among the layers.

Challenge 2: Limited context information of small objects. Liu et al., 2021.

Small objects have little feature information.

Challenge 3: Class imbalance for small objects.Liu et al., 2021.

Many anchors are generated which is an issue.

Challenge 4: Insufficient positive examples for small objects. Liu et al., 2021.

Insufficient examples.

There are many ways to get over few of these hurdles as said in Pham et al., 2017 but it is still not sufficient as the vortex core is extremely small. From the experiments done by Pham et al., 2017 and Nguyen et al., 2020, it is seen that YOLO, SSD and Faster RCNN are the best models for this use case.

YOLO works by framing object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. A single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on detection performance as stated by Redmon et al., 2016. This architecture is built

for speed and fast detection. It is not as accurate as Faster RCNN but is sufficient to accurately locate vortices. YOLO also struggles with detecting small objects in flocks or groups. This model can sometimes struggle to generalize as stated by Redmon et al., 2016.

The main feature of a RCNN model is the region proposal network, this network takes an image as input and outputs a set of rectangular object proposals, each with an object score as stated by Ren et al., 2015. The loss function for an image is then calculated the loss is as follows.

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*).$$

Figure 9: Loss function for RCNN)

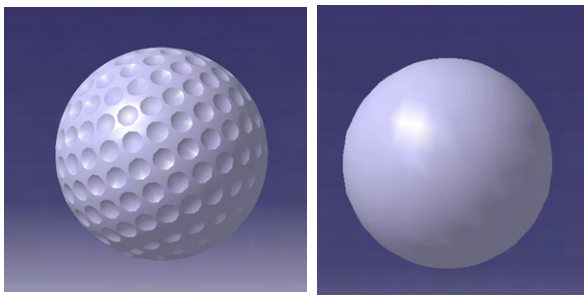
Here, j = index anchor, pi = probability of predicted anchor, ti = a vector representing the coordinated of the bounding box, Lcls = classification loss, Lreg = regression loss, Ncls = normalized value of pi, Nreg = normalized value of ti Ren et al., 2015.

Vortex detection is an important aspect of designing aerodynamic systems. Traditional vortex detection methods like numerical calculation are accurate but slow, according to Yanyang et al., 2020 CNN based object detection models performed better compared to the traditional methods. YOLO was used to compare. It will be interesting to see if better results can be obtained than the results produced by Yanyang et al., 2020.

# 3 Methodology

## 3.1 CFD

### 3.1.1 CAD



(a) (b)

Figure 10: CAD Models of (a) Golf Ball (b) Smooth Ball

The above figure shows the CAD model of Golf Ball and Smooth ball, designed using CATIA, the radius of the Golf Ball is 21.082 mm, which is the radius of the real Golf Ball. The dimple radius is 2.26 mm. The radius of the Smooth Ball is 21 mm, the radius of Smooth Ball is approximately the radius of the Golf Ball. The Smooth Ball is designed close to Golf Ball size for validation purposes.

| Geometry | Radius |
|---|---|
| Golf Ball | 21.082 mm |
| Dimple | 2.26 mm |
| Smooth Ball | 21 mm |

Table 1: Dimensions of Golf Ball and Smooth Ball

### 3.1.2 Domain

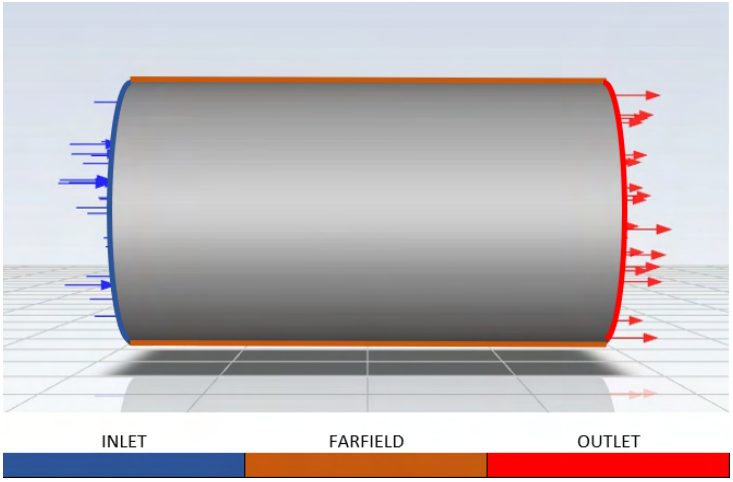


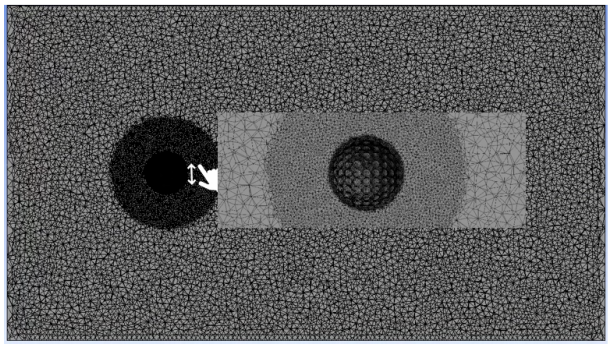

Figure 11: Domain used for Simulation

The Domain in above figure is used for solving the fluid flow problem, before choosing this cylindrical domain, simulations were performed using rectangular, symmetrical domains. The Domain was designed using SpaceClaim.

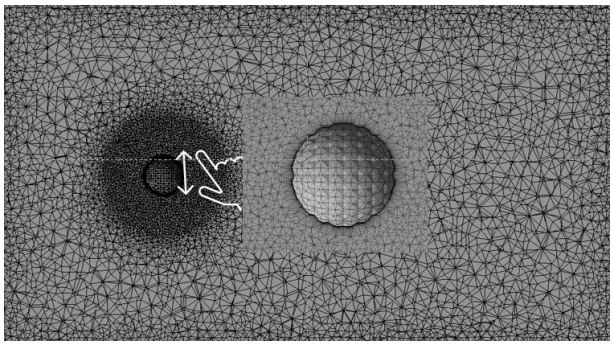
| Cylinder | Length |
|---|---|
| Height | 640 mm |
| Diameter | 360 mm |

Table 2: Dimensions of the Cylindrical Domain

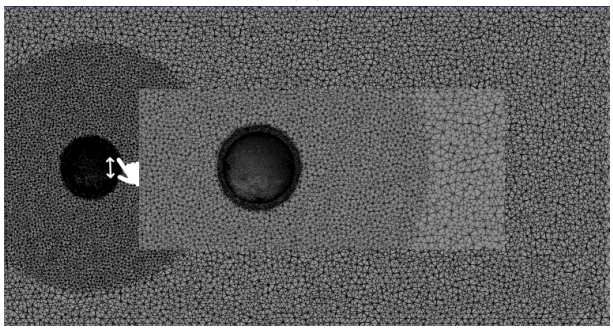
### 3.1.3 Mesh



(a)



(b)



(c)

Figure 12: (a) Fine Mesh of Golf Ball (b) Coarse Mesh of Golf Ball (c) Fine mesh of Smooth Ball

To mesh the domain, Mesh workbench from ANSYS is used. To have fine elements around the Golf Ball, Body Sizing method is applied with Sphere of Influence option to capture the vortex and Inflation Layers are used to capture the flow separation and boundary layer better. Body of Influence could have also been used to capture the wake region but

using body of influence would increase the fine elements in the mesh, hence increasing the computational difficulty of the problem and increased computation time. Since the main objective of the simulation is to capture the vortex and to calculate Lift Coefficient and Drag Coefficient with at most accuracy. Using Sphere of Influence is efficient enough because the mesh region surrounding the vortex formation is fine and the golf ball mesh is also fine with inflation layers, to calculate Lift Coefficient and Drag Coefficient better and to capture flow separation. Similar settings are used to mesh the coarse mesh. The Coarse Mesh has 0.85 Million Elements whereas the Fine Mesh has 5.2 Million Elements. The coarse mesh has less fine elements as that is evident from the above figure. Similar settings are used to mesh the Smooth ball.

### 3.1.4 Simulation Setup

Pressure-based Transient solver is used for this simulation. Pressure-based solver is used to solve for incompressible and mild compressible flows, which is ideal for this problem. For the viscous model, K-Epsilon, K-Omega, Transition-SST, Transition K-Kl Omega, Reynolds Stress models were used to compare the results.

| Conditions | Values |
|---|---|
| Velocity | 45 $ms^{-1}$ |
| Reynolds Number | $1.27 \times 10^5$ |
| RPM | 2000 - 4000 |
| $\alpha$ | 0.2° - 0.4° |
| Projected Area | 0.0016123 $m^2$ |

Table 3: Boundary Conditions

To Simulate a rotating Golf Ball, Golf Ball is defined as moving wall in boundary conditions, rotational motion is chosen, rotational axis and the speed of rotation is inputted RPM.

Inlet is defined as Velocity-Inlet with 45 m/s as the inlet velocity. To change the angle of the Golf Ball, the flow angle is resolved to replicate the change in angle of the golf ball by fixing the golf ball angle.

Outlet is defined as pressure-outlet with default values, pressure-outlet is usually used to define the static pressure at flow outlet, the gauge pressure at outlet is set to 0, defining that there is no pressure more than the local atmospheric pressure.

Initially the simulation is performed for non-rotating golf ball to capture the vortex and share the vortex image with the AI team. At Upon completion, the finalized domain and mesh is simulated for rotating golf ball, varying angle of attack, varying viscous models etc.,

During the initial phase of the simulation, there were numerous problems encountered, problems like:

- Floating Point Errors
- Fluent Ran out of Memory

- Having the simulations run 24x7, the most frustrating problem encountered was "VMware session timed out" which is losing connection to the VMware server and having to restart the simulation from the beginning
- Trying to validate CL and CD values but due to insufficient information on the area used to validate the results in () report so many different simulations were performed on different meshes, different domain types and sizes, but later it was found out by trial and error that it was projected area which was used to calculate CL and CD instead of wetted area.

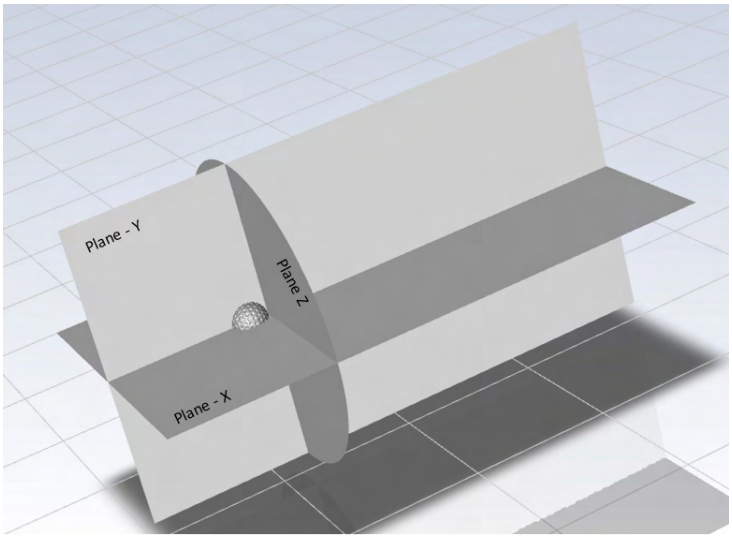



Figure 13: Positioning of Planes to run macros to write results

In Order to export the results from Fluent at every time-step to ParaView macros were defined in Fluent and solution at every time-step is saved for each plane. This is saved as '.plt' file which is tecplot file. The results are exported from 3 different planes, planes are defined perpendicular to XY plane, YZ plane, XZ plane. This file is later open in ParaView and Surface LIC plugin is used to visualize the vortex, Surface LIC is Line Integral Convolution, it normalizes the 3-D component of velocity and helps us visualize the 3-D flow in 2-D.

## 3.2 SE

The method we used for the software development is at first glance based on agile methodology as the milestones is broken into two sprints, and that within each sprint tasks are also broken into short that way changes in project requirements are easily addressed with little impact on the planning. Indeed, as we didn't know for sure the technologies to use or how to implement a user interface with a trained machine learning algorithm we needed to self-research and find how to proceed. In fact, we started out to use Django as our first python framework. Django seemed to be appropriate for our final product, the idea was to make the ML algorithm available by REST API. This method requires having a running server which handles requests and forwards them to ML algorithms. Furthermore, Django comes with a built-in database backends which could have saved us some time on the database configuration. We spend around a month on that particular framework as we didn't have a working accurate machine learning algorithm at the time,

and had a working deployed models with Django which only worked for a specific example as the implementation was based from a tutorial.

Moreover we also decided with certainty to work with react JS as it is one of the best JavaScript framework to build user interfaces that also allows us to take reusable user interface components from the web .Also, because it allowed us to use JSX to quote HTML and uses the syntax of these HTML tags to display components.We then had a first look of our user interface deployed on the web thanks to Heroku, a container-based cloud platform as a service, we decided to use it as it is a free service and not to worry too much about configuring and maintaining servers.

The goal of our first sprint was to have a functional web-application using machine learning algorithm in the back-end, even inaccurate one. The artificial intelligence team provided us with a working application however they used a different framework than us, Flask. They found it more flexible and easy to use, after some discussions and since they are assigned to write the Api were we would fetch the data, we decided to use Flask.

Second part of the project from on SE perspective, after finishing successfully the first sprint continued with agile development methodology were the stages of development are organized into a series of cascading procedures.First one feature definition, development and implementation then second feature etc. We also went for a development similar to a MERN stack that is a set of dependable and powerful tools for building scalable master web applications, and also because it is made up of four JavaScript-based technologies :

- MongoDB:a document-oriented database,that has more flexibility compare to SQL databases.

- Node.js :a runtime environment for JavaScript that allows you to create network applications.

- Express.js : a framework based on Node.js used to create APIs

Guidance on each button or link is detailed in the user manual in table 15.

## 3.3 CIDA

### 3.3.1 Introduction to Object Detection

Data on the airflow around the golf ball was used for Computer Vision. In the method of detecting the vortex from the continuous time-series data of the airflow confirmed through this, the object detection technology using the deep learning technique was applied. Object detection refers to the task of setting an area through a bounding box in a specific image and determining whether an object exists or the type of object within the area. Various technologies have been developed in this method, and various technologies have been developed based on the concurrent neural network deep learning technology.

Once the technology was first developed, the focus was on the accuracy, but with the advent of autonomous vehicles, speed also became an essential point. Although RCNN (Recurrent and Concurrent Neural Network)-type models have a certain level of accuracy,

they are not fast enough to be applied in a real field. Recently, several neural networks such as YOLO have been developed that increase the speed of object recognition while maintaining a certain level of accuracy. YOLO was also applied to the technology for this task, and the results were compared based on various versions and datasets.

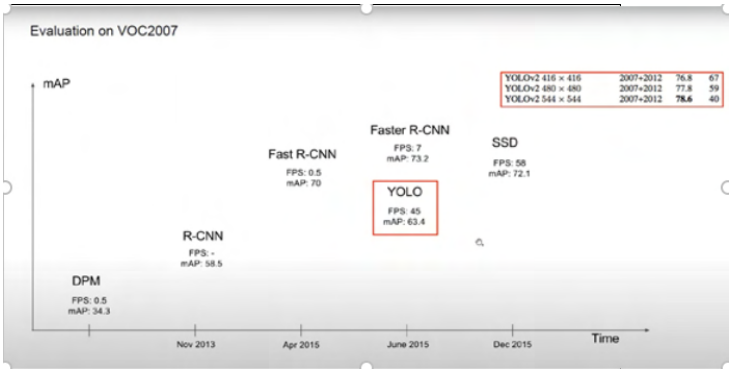



Figure 14: Comparision of different object detection models on accuracy

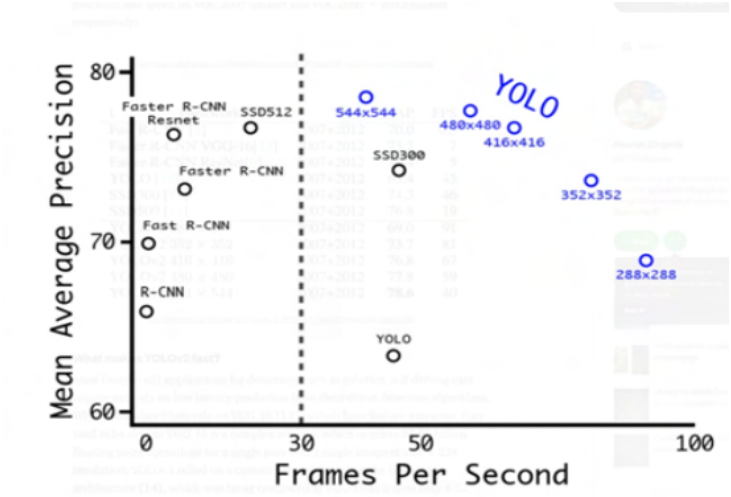



Figure 15: Comparision of different object detection models on speed

* mAP (Mean Average Precision) * FPS (Frame Per Second)

### 3.3.2 Object Detector Architectures

There are two types of object detection models -

**One stage detectors** – Models having this kind of architectures form a grid on the images and calculating the grid probabilities of having a piece of objects within them while also calculating the class probabilities. The bounding boxes are then drawn on the grids, these predictions are then filtered using non max suppression (A filtering method using Intersection Over Union). As all of this happens in a single stage, models with this architecture are very fast and fully convolutional (any size image can be the input). There are two object detection models using this architecture YOLO and SSD. YOLO is explained below.

**YOLO (You Only Look Once)**

This model reconstructs object detection as a single regression problem, from image pixels to bounding box coordinates and class probabilities. Accordingly, a single convolutional

network has completed an integrated model that predicts multiple bounding boxes in the entire image and calculates the class probability in each box at the same time.

First, the structure and operation principle are explained based on the contents of the YOLO v1 paper published in 2016 CVPR (Computer Vision and Pattern Recognition), and the changes as the version evolved are briefly explained. Figure 1 shows the process of the YOLO model. Once an image is an input, the image is divided into a grid (7x7) and can be divided into two processes. One is about the bounding box. If the grid includes the centre of the bounding box, the response value is activated in the grid. This includes the location information of the bounding box and the confidence score. Another is for classification. Each grid calculates the class probability, and each grid has a probability value for the classes. After that, the final result value is derived through NMS (Non-max suppression, an algorithm that removes boxes in similar positions and selects the most suitable box).

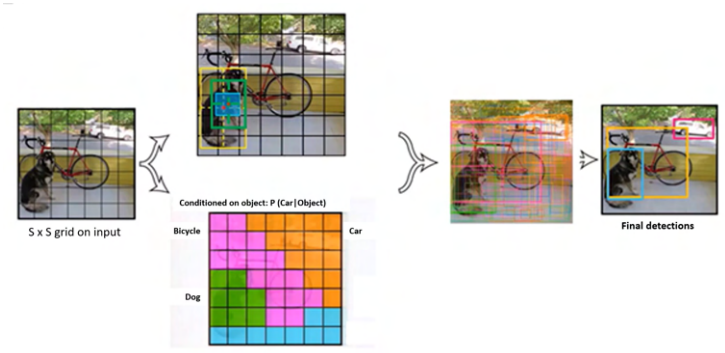



Figure 16: Object Detection method of YOLO model
https://pjreddie.com/darknet/yolov1/

The neural network structure is a structure made based on GoogleNet and is implemented with a total of 24 convolutional layers and two Fc layers. (Figure )

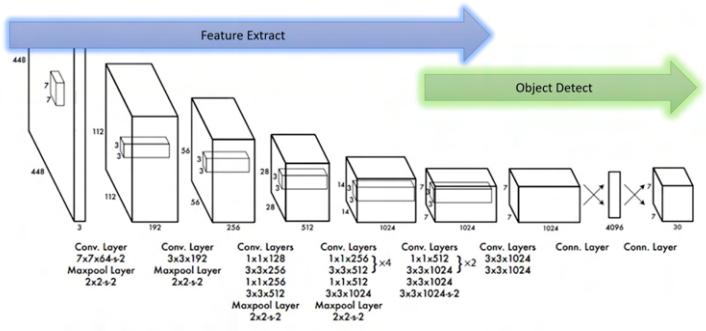



Figure 17: Different layers of a CNN model

The 7x7x30 matrix derived from the neural network is composed as shown in Figure 3. The information in the matrix(tensor) is the bounding box information and the probability value for the class of each grid.

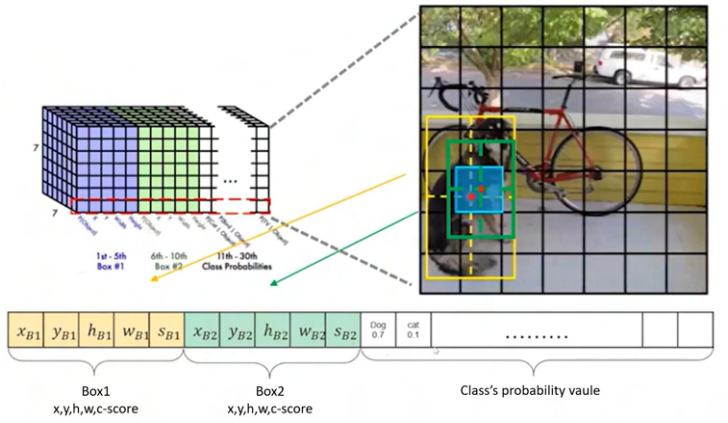



Figure 18: YOLO working

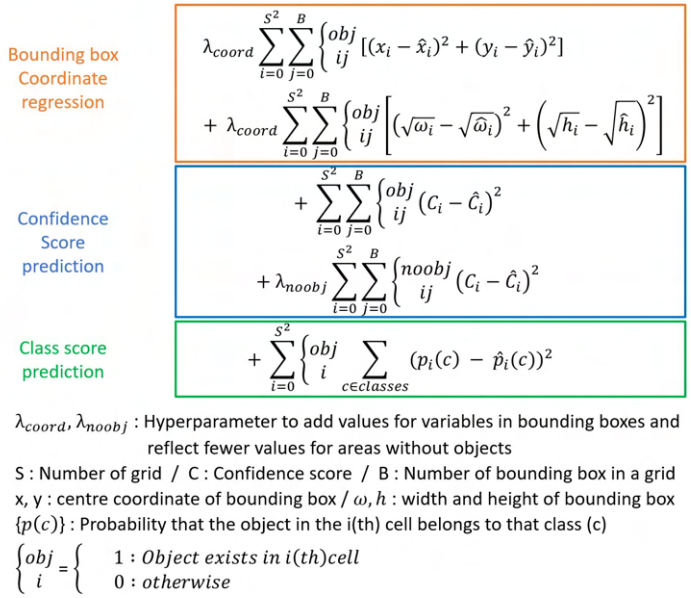



Figure 19: YOLO equations

Since YOLO approaches the object detection problem from the point of view of a single regression problem, a Loss Function that combines the values of Bounding box Coordinate regression, Confidence Score prediction, and Class score prediction as above is implemented.

Different versions of the YOLO models are explained below :-

1. YOLOv2

   - Batch Normalization: Apply batch normalization to all convolutional layers
   - Remove drop-out by obtaining regularization effect through batch normalization - Performance improvement of about 2% based on mAP
   - High Resolution Classifier: Learning the classifier while maintaining the image resolution of the detection task without reduction - About 4% performance improvement

- Convolutional Anchor Boxes: Remove the FC layer and use the Fully Convolution structure - Predict the bounding box using Anchor Box.

- Fine-Grained Features: In order to detect small objects more, the layer from the previous stage with higher resolution is taken and concatenated to the output feature map for detection.

- Convert the feature map of 26 by 26 by 512 to the feature map of 13 by 13 by 2048 and use it - About 1% performance improvement

- Using DarkNet: Converted from VGG-16 based detection frameworks to a network structure called DarkNet - Convolution layer (19), maxpooling layer (5).

2. YOLOv3

   - Apply deepen the DarkNet network structure to convolution layer (19 - 53) - the overall accuracy (especially small object detection) is increased

3. YOLOv4

   - Based on v3, the main purpose on increasing accuracy.

   - Apply existing technologies (Style transfer GAN, Random erase, Spatial Dropout, GIoU, Feature Pyramid Network, Leaky ReLU, etc.) to training phase and inference phase.

   - Input resolution 512 x 512 applied (Improvement of small object detection) - Overall, YOLOv4 applied CSPDarknet53 to improve classification performance based on YOLOv3, applied BoF and BoS to pre- and post-processing, respectively, and applied PANet (Path Aggregation Network) to the neck part.

**Two stage detectors** - Models having this type of architectures have two stages, the first one is passing through the Region Proposal Network, here regions of interest are generated, these regions having a high probability of having an object within them. Then in the second stage, these regions are passed through classification network where they are classified and then the results produced. Some of the models having this architecture are Faster RCNN, RCNN, FPN etc. The architecture of RCNN is explained below.

**R-CNN (Region-based Convolutional Neural Network)**

The 2-stage architecture is a method in which classification and localization are separated. This is a typical object detection method using CNN (Convolutional Neural Network) technique.

The R-CNN model selects about 2,000 candidate regions through a Region Proposal algorithm called Selective Search, and performs classification and bounding box regression with these regions as input. It was advantageous to be able to classify the classes of each Region using the CNN technique. However, the entire framework could not be trained end-to-end. Therefore, there was a disadvantage in that it was difficult to find a Global Optimal Solution. In addition, it required a long learning time by performing CPU-based

Selective Search and the large storage space required was also a disadvantage. (Individual image processing time: 47 seconds)

Fast R-CNN improved from R-CNN reduced individual image processing time after that. Fast R-CNN was able to learn by grouping all of the Feature Extraction, RoI Pooling, Region Classification, and Bounding Box Regression steps as End-to-End. But still, the first, Selective Search was performed on the CPU, so it was slow.

An improved model for this was Faster R-CNN. By proposing RPN using CNN technique applying GPU, the entire framework could be rapidly trained end-to-end. However, it was still composed of many components, and each feature vector was individually forwarded to the FC layer in the region classification step.

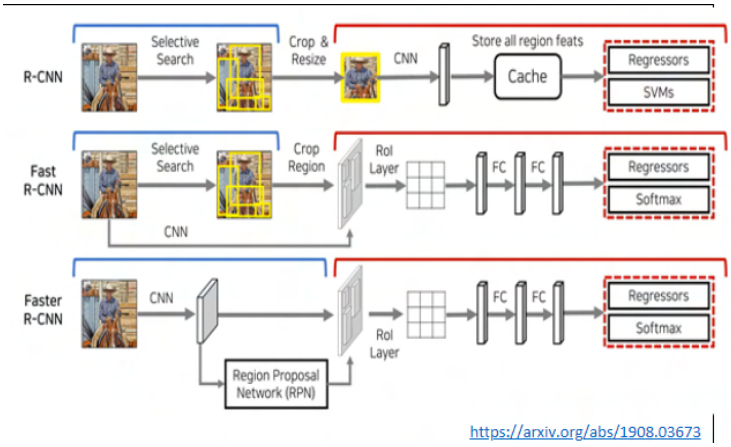



Figure 20: Region-Based Convolutional Neural Network working

### 3.3.3 Training and Evaluation of Different CNN Models

In this section, the method for training, evaluating and comparing different CNN models are shown. The object detection model used in this study is the YOLO v5 based on 1-stage architecture, but a comparative test was conducted with the Faster R-CNN model based on 2-stage architecture to select the YOLO v5 model. In this regard, the composition and operation principle of the YOLO model will be explained including a brief description of the Faster R-CNN model, which is a representative model of the 2-stage architecture. First, the evaluation index related to the assessment of the object detection model will be briefly described.

**Performance evaluation**

| | | Predict Result | |
|---|---|---|---|
| | | Positive | Negative |
| Ground Truth | Positive | TP (True Positive) | FN (False Negative) |
| | Negative | FP (False Positive) | TN (True Negative) |

Figure 21: Confusion Matrix

The following metrics are used to evaluate the trained model on a validation or a test dataset.

- Accuracy : TP + TN / TP + FN + FP + TN

- Precision : Number of correctly detected objects (TP) / Number of objects detected by the model (TP+FP)

- Recall : Number of correctly detected objects (TP) / Number of truth answer objects (TP + FN)

- F1 - Score (Harmonic mean of precision and recall) : 2Precision  Recall / Precision+Recall

- mAP (mean Average Precision) : Average of each class's AP (Average Precision)

**Dataset**

2000 images representing different views of vortices were used to train the YOLO v5 model. Different views were included so that the model is generalized to different images of vortices. These images were a result of CFD simulations of fluid flow over golf ball.

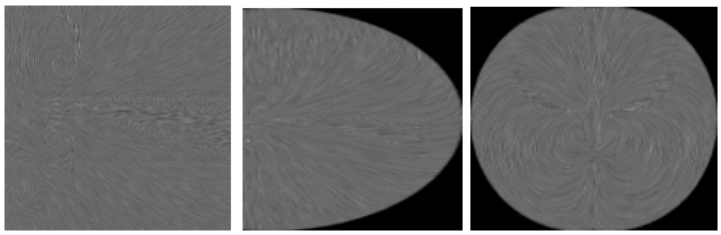


Figure 22: Simulated images containing vortices, shown in different views.

**Training**

The learning method of the object detection model is similar to the learning method of the regression model. It learns iteratively and tracks low loss values. Coordinate values and the object's Probability values are used here. The epoch value applied for training is set to default 1000. If an improvement value is not derived during operation for a certain period of time, learning is stopped. Additionally, the image size used for training affects learning speed, memory usage, and more. Resizing to a small image is helpful for cost (time consuming), but inputting a large size is helpful for detecting small objects such as our vortex. In this test, our model is tested by setting it to 640x640.

**Hyperparameter Evolution**

In machine learning Hyperparameters control the training process of a model. Tuning these parameters can produce varied results in the model performance. Using traditional methods like grid search or random search can be computationally expensive and can take a long time. The following steps are a better option when performing hyperparameter tuning in YOLOv5.

1. Initialize the parameters, or use default values which are trained on COCO dataset (there are 30 parameters)

2. Define fitness – It is the evaluation metric that needs to be maximized. Fitness = [ Precision, Recall, mAP@0.5, mAP@0.5:0.95 ] for this model, the fitness was defined

as shown. Fitness =[ 0, 0, 0.2, 0.8 ] this means that precision and recall do not matter while tuning, mAP@0.5 has 20

3. Evolve – Using the command below, start the evolution process to tune the hyper-parameters.

4. Run the evolution using the following command python train.py –epochs 10 –data myconfig.yaml –weights yolov5m.pt –cache –evolve

### 3.3.4 Experimental approach to model selection

The method of selecting a model in this study is comparison through experimentation. The first compares the one-stage detection method and the two-stage detection method. Since our task is to find the vortex in the air flow field, we need to consider the image shape of the video clip. Therefore, the detecting speed is the criteria for this test. The second is to verify the reliability of the model. It's about making sure that the trained model shows the results we expect. This is to check the response related to any improvement method (increasing training data) is applied to the model. The third is accuracy improvement. It is normally understood that the neural network of the object detection model shows higher accuracy as its depth increases. Therefore, after learning models with different number of layers, we select a model suitable for our task. The above is an experimental approach for basic model selection, and methods for better model selection may be added.

## 3.4 SIFT Features for Vortex Detection

In this section the process by which we aimed to develop an alternative to CNNs for vortex detection in an image. The method we proposed aims to use the edge information within an image to classify regions into regions that contain vortices and regions which do not.

The method we used is based on the concept of the Scale Invariant Feature Transform (SIFT) introduced by Lowe, 2004. For a given image, we use SIFT Keypoint descriptors to create descriptive features to classify images using a simple Neural Network. When applied to an actual image, we pass a sliding window over the image to determine approximate boxes as to where we believe the vortices lie.

### 3.4.1 Dataset Preparation

To train the networks we would use the labelled vortex regions to produce two different datasets. One with a collection of images sized $64 \times 64$ and one with a collection of images sized $32 \times 32$. The generation of the data within each dataset was split into two parts, the generation of the vortex images and the generation of the non-vortex images. The non-vortex images are generated by passing over a sliding window over the image and checking whether a there is a manually labeled vortex within this region. If there is, we ignore this box and move onto the next otherwise we add the current box's position to the dataset for non-vortex regions.

The Vortex images are generated using a method similar to that described by Zeiler and Fergus, 2013. In the paper they describe that for each image, after they have been resized,

10 sub-crops of the image are taken and are added to the dataset. For our process, for each of the manually labelled vortices we generate 10 randomly taken 'sub-crops' where the centre of the image is adjusted by a random amount in both the x and y directions. The random amount of shifting was limited such that at least half the labeled vortex would remain in the image generated. Figure 23 displays some of the examples in the dataset.

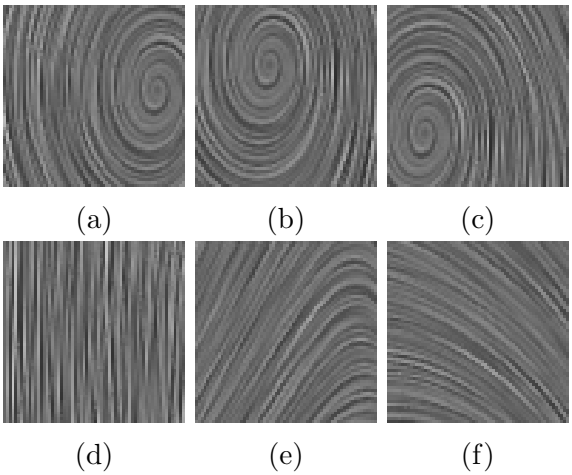



Figure 23: Some example images generated for the $64 \times 64$ dataset. (a)-(c) display some of the images generated for a single vortex, while (d)-(f) show the images generated in the non-vortex regions

We applied this method to 700 images to generate the two datasets. After these images were generated we ended up with 13330 images of vortices and 60048 images of non-vortices for the $64 \times 64$ dataset and 259897 images of vortices and 13330 images of non-vortices for the $32 \times 32$. This means that we end up with 51365 images in our training set and 22013 images in the validation set for the $64 \times 64$ dataset and 191259 in the training set and 81968 in the validation set for the 3232. This method does however mean that we end up with a much larger number of images without vortices than we do vortex images for both sets of data.

### 3.4.2 SIFT Keypoint Descriptor generation

The idea for using SIFT features for within this model was based on the appearance of vortices in the images and the principle behind why they appear like this. When a vortex occurs in an image, we tend to see spiral or circular patterns forming in that region as shown by Figures 23(a)-(c). The idea is that when we generate a map of the edge directions of the image of a vortex, in theory we should get a large amount of variance in these edge directions. So when we generate a SIFT keypoint descriptor covering the $64 \times 64$ or $32 \times 32$ regions generated in the dataset preparation, we should find the descriptor for a vortex region to be vastly different for a region without a vortex such as those in Figures 23(d)-(f).

To generate the edge responses at each pixel location we apply the convolution operation, as described by Szeliski, 2022, over the images using the Sobel kernals as defined by Sobel, 2014. The Sobel $x$ and $y$ Kernals shown in Figure 24 are the Kernals applied to the images to compute the edge responses in the horizontal and vertical directions. Once we have computed both responses we can compute the direction and the magnitude of the edge

| | | |
|---|---|---|
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

(a) $x$ Directional Sobel Kernal

| | | |
|---|---|---|
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

(b) $y$ Directional Sobel Kernal

Figure 24: The $x$ and $y$, $3 \times 3$ Sobel kernals, computing the magnitude of the edge responses in the horizontal and vertical directions respectively

response at each pixel location using simple trigonometry:

$$Magnitude = \sqrt{x^2 + y^2}$$

$$Direction = tan^{-1}(\frac{y}{x})$$

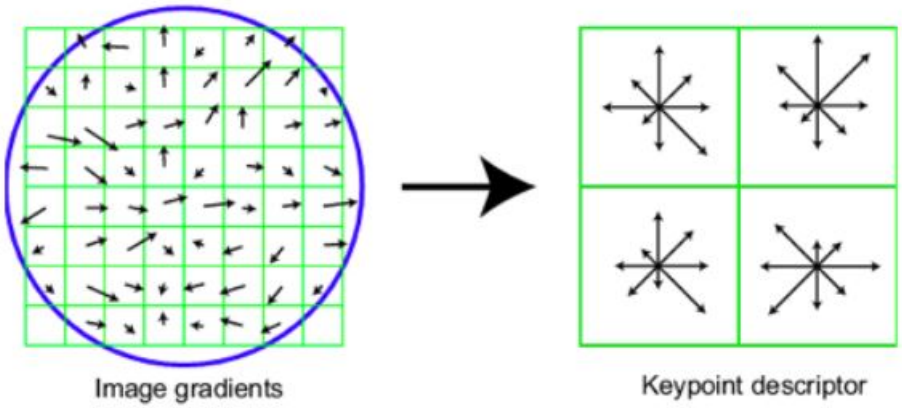



Figure 25: A visual representation of the computation of the SIFT Descriptors from Lowe, 2004

Once we compute this information for each image a SIFT descriptor can be computed for each image. We define a number of regions to describe the image by and summarise the information within each region by computing a orientation histogram. Figure 48 shows exactly what is meant by this. On the left we have the computed gradients and orientations, whilst on the right we have the final descriptor. For each 4×4 box a histogram of the summarised orientations is computed. The histogram consists of 8 boxes with increments of 45°. For this project we only used 8 boxes in the orientation histogram and each pixel value was added to the nearest histogram box. In addition the descriptor is computed over the entirety of the 64×64 and 32×32 images. Once the feature is generated we also normalise the response in each region to prevent any bias towards strong edge responses.

### 3.4.3 The Proposed Model

The final model ended up being a relatively simple Neural Network, with an input and an output layer with two hidden layers. The size of the input layer is dependent on how

many regions we define for the SIFT features, as for each region we define 8 histogram boxes. There are two output layers, one for the classification of an image as a vortex and one for the classification as a non-vortex. Since the problem aimed to test the viability of different SIFT features on vortex classification, we also used the same number of hidden layers for each variation of the model. For the first layer there were 64 nodes and there were 32 for the second layer.

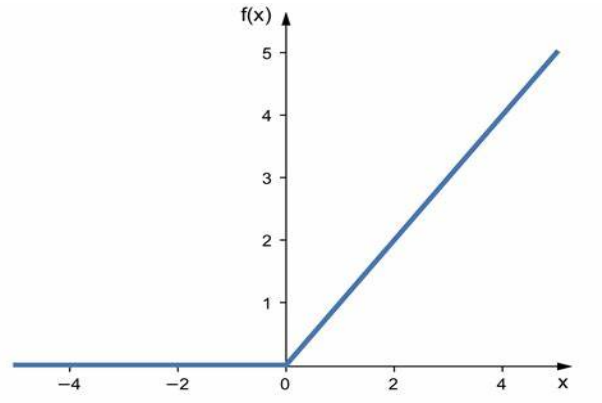



Figure 26: The ReLU activation function

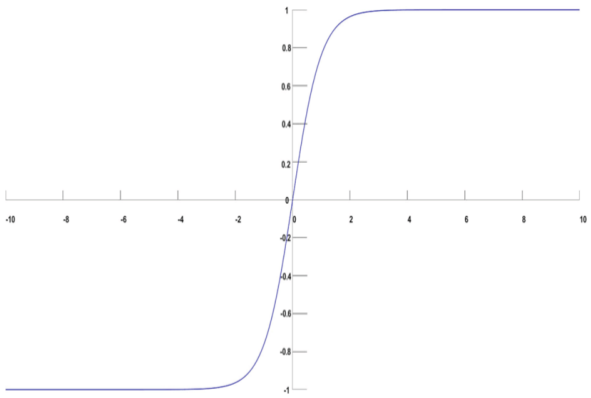


Figure 27: The Softmax activation function

On each of the hidden layers we apply the ReLU activation function as defined by Nair and Hinton, 2010 which is displayed in Figure 26. Since we know we will not have any negative responses, this function was chosen. On the output layer we chose to use a Softmax activation function, as described by Goodfellow et al., 2016, since we wanted to produce a probability distribution describing the likelihoods of an image either containing or not containing a vortex. In addition, the function is minimised using the binary cross entropy since the classification for each entry is either a 1 or a 0. The binary cross entropy is defined as:

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}(y_i \cdot \log(\hat{y}_i)) + ((1 - y_i) \cdot \log(1 - \hat{y}_i))$$

Where $n$ is the number of data samples, $y$ is a vector of the true labels and $\hat{y}$ is a vector of the predicted labels.

For the optimization process the 'Adam' algorithm was used, as defined by Kingma and Ba, 2015. To evaluate the performance of the algorithm we recorded the accuracy, precision and recall. Finally, we performed 500 epochs for each method and a graph is plot of

the performance of the algorithm over the validation data on each epoch for each metric. The final performance of each algorithm is also recorded and displayed in this report.

## 3.5 Proposed method for vortex core finding

To find the core of a vortex we propose an algorithm based on the concept of Gradient Descent over a 2-dimensional function. Lemaréchal, 2012 explains the concept of Gradient Descent in more detail, but the important concept behind this algorithm is that by descending a function in the direction of the gradient, we can identify local minima in a function.

First of all, to calculate the perceived gradient in an image we can apply the Sobel operators described in section 3.4. By calculating the magnitude and directional responses using these operators we can produce an estimate to the gradients of an image based on the edge responses. Even though the vortices within an image do not tend to be minima, due to their circular and spiral-like nature, we find that the edge responses tend to point themselves towards the vortices and thereby their centres.

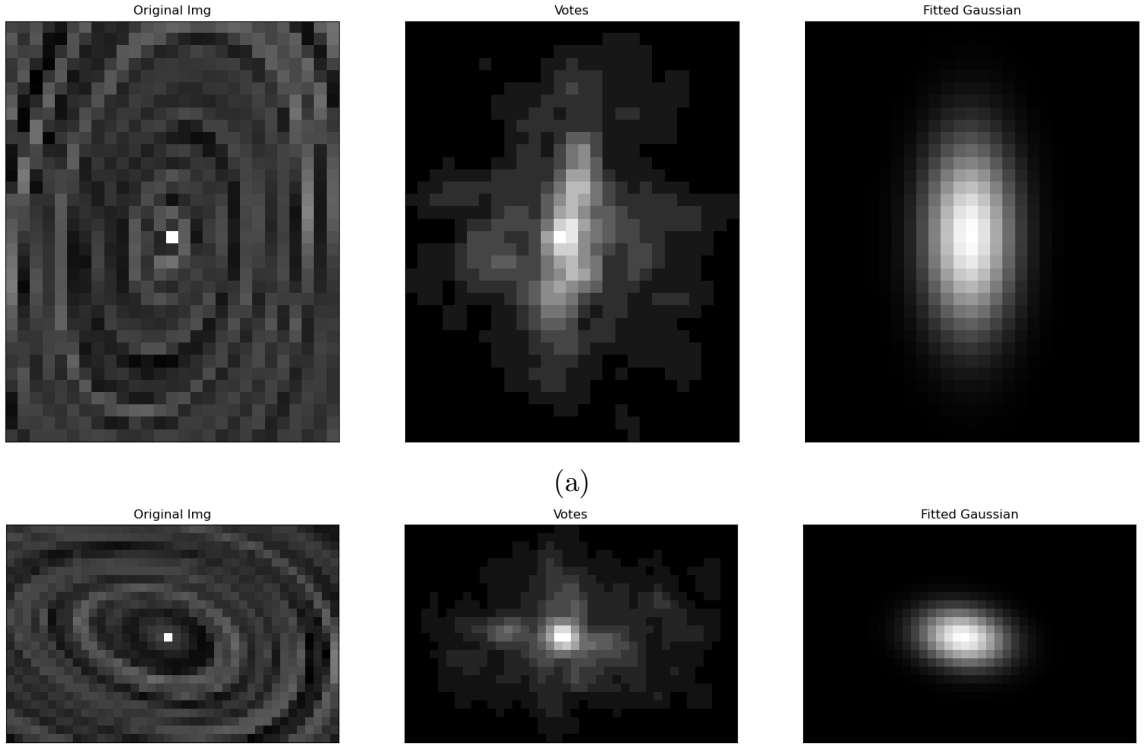



Figure 28: Images displaying the Original Vortex, The probability Distribution Generated and the fitted Gaussian Distribution respectively

To actually identify the precise core location we proposed the algorithm represented in pseudocode in Algorithm 1 in Appendix I. The algorithm first calculates the gradient directions for every point in the vortex image and then begins the voting stage. During this stage we perform a number of gradient descent iterations to vote as to where the core is believed to be located. For a number of steps, we move a random amount in the direction of the Sobel edge responses. If the vote moves outside the dimensions of the image we terminate the process and try the next vote. In addition, since we do not know if the edge is pointing towards or away from the centre of the vortex, we randomly choose

to either move in the direction indicated by the Sobel operators, or in the direct opposite direction. After the steps have completed, we add a vote to the final location and repeat the process for the specified number of iterations.

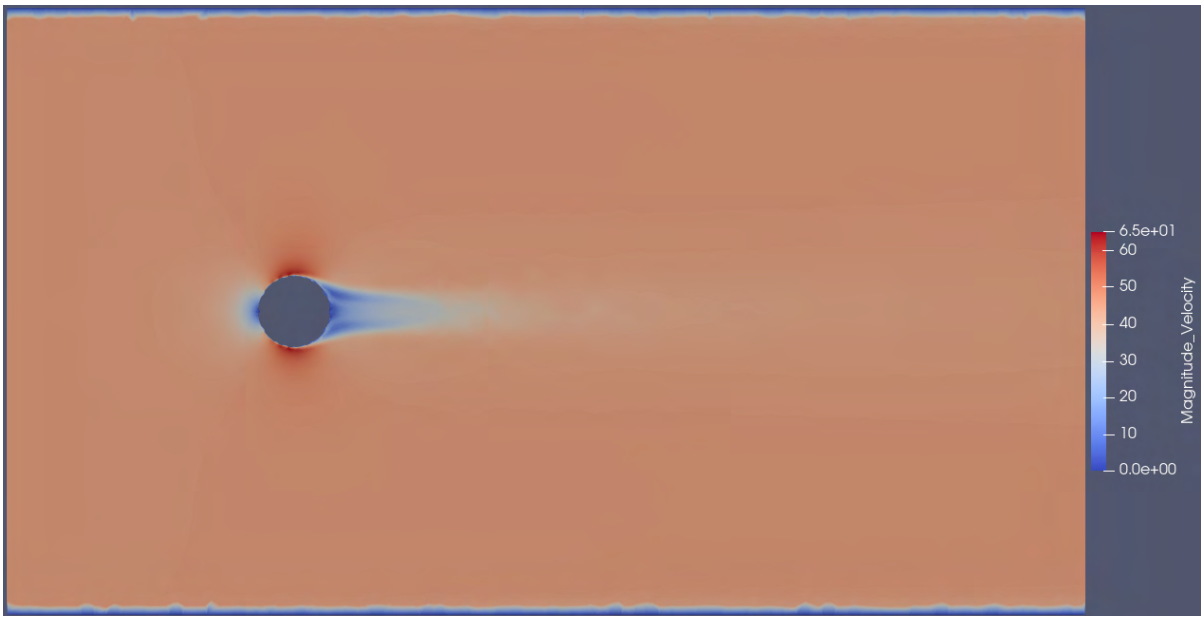
What we end up with is a probability distribution as to where the core is believed to be located. We chose to fit a Gaussian distribution to the final outcome to attempt to approximate the shape of the resulting vortex. By doing this we estimate the orientation and width in the x and y direction and we take the centre of the gaussian distribution to be the final core location. Figure 28 displays two examples of the process. The left images show the original vortex image and the predicted core location, the middle image show the probability distribution generated and the right images show the fitted gaussian distribution used to estimate the core location.

To compare this method, we have manually labelled the cores of 700 vortex images and will produce a mean error represented as the pixel distance between the predicted and labelled location. In addition this method will be compared against the benchmark error of purely using the centre of the bounding box as the core for each vortex.

# 4 Results, Discussion and Validation

## 4.1 CFD

### 4.1.1 Results




(a)


(b)

Figure 29: Velocity Magnitude Contour of (a) Golf Ball (b) Smooth Ball

The post-processing was performed using ParaView. Fig.29 is the velocity magnitude contour of rotating golf ball and smooth ball at 4000 RPM at an Angle of Attack of 0.2°. As the flow flows over the golf ball, the flow can be seen separating at an earlier stage for golf ball, whereas the flow separation happens later for smooth ball. The flow separates at an earlier stage because it paradoxically creates small turbulent regions near the dimples as the surface is uneven. The fluid momentum is closer to the ball, as the flow flows over golf ball, the early flow separation helps the flow to reunite quicker for golf ball than

smooth ball. resulting in lower Drag Coefficient since the flow reunites quicker the wake region is smaller, hence less turbulent flow in the wake region with respect to smooth ball's wake region. Lower Skin Friction Drag, Form Drag and Induced Drag due to less turbulent flow in the wake region for Golf Ball, Hence, lower the drag coefficient for Golf Ball than Smooth Ball.

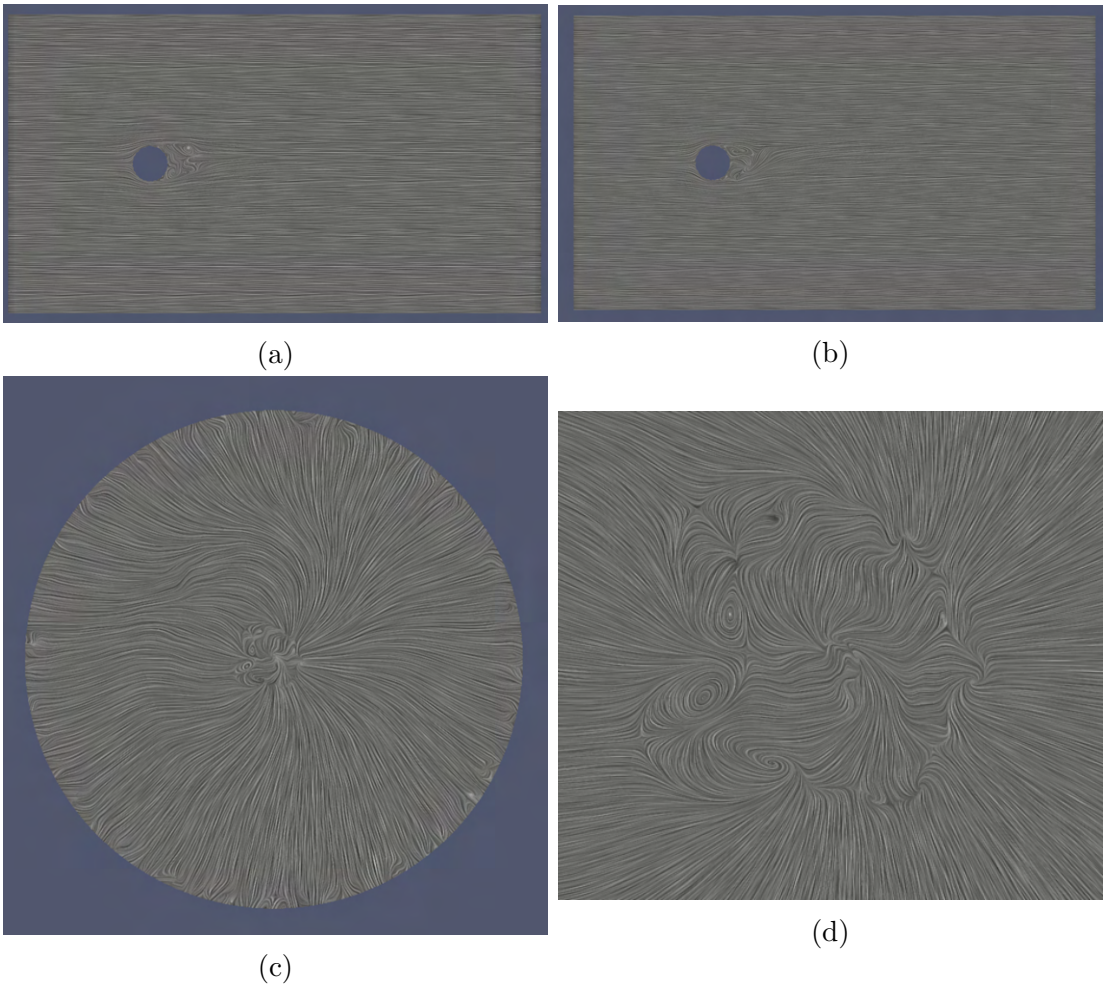


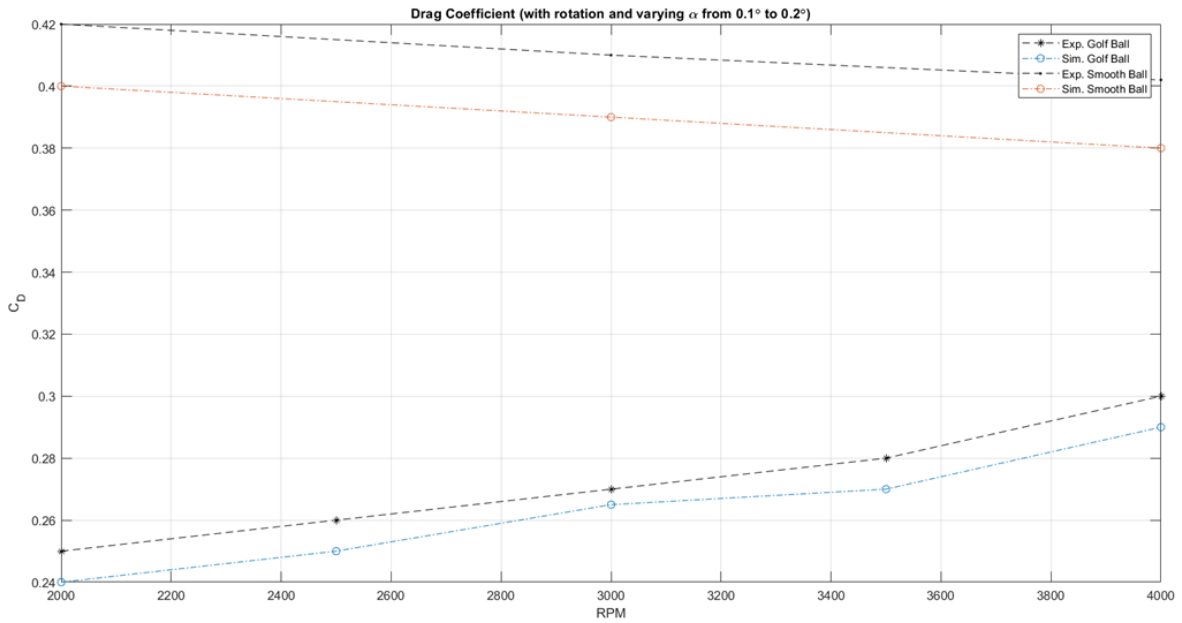

Figure 30: Visualisation of Vortex using Surface LIC plugin on (a) Plane X data (b) Plane Y data (c) Plane Z data (d) Zoomed view of Plane Z

Fig.30 shows the visualization of vortex using Surface LIC plugin in Paraview, Surface LIC plugin normalises the 3D vector and helps us visualise the 3D flow in 2D plane. The vortex images are exported for 3 different planes (as shown in Fig.13) for 200-300 timesteps, in the end 600-900 images are exported from one simulation. These vortex images are later used to detect vortex using YOLO.

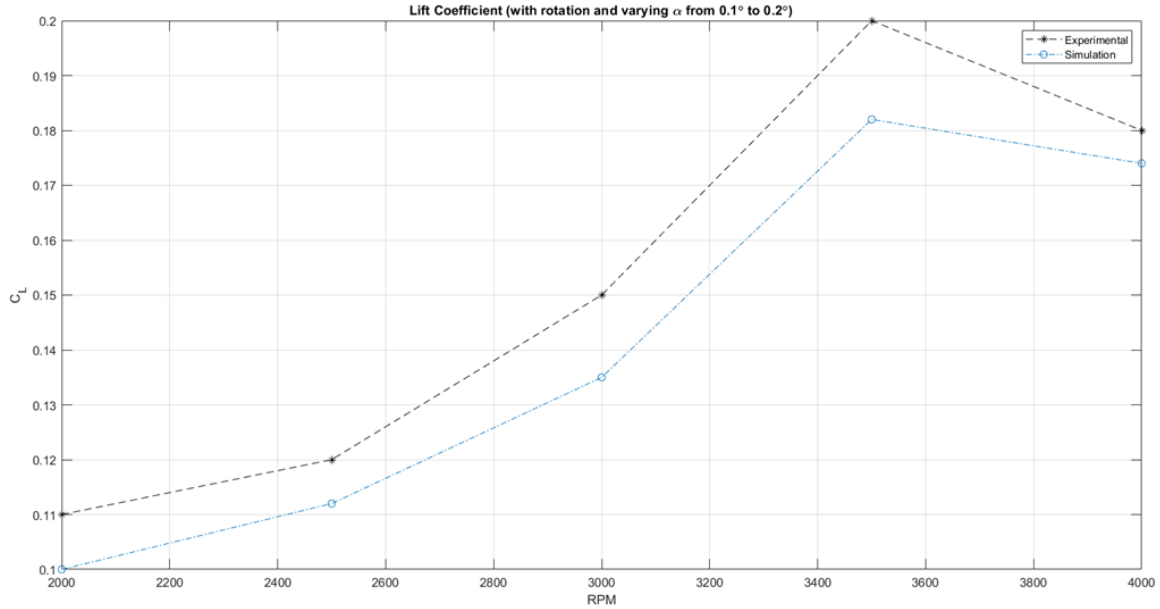
### 4.1.2 Validation

Fig.31 is plotted for Drag Coefficient, for different RPM and $\alpha$. RPM is varied from 2000-4000 with increments of 500 and $\alpha$ is varied from 0.1° to 0.2° with increments of 0.025°. The results are obtained for Golf Ball and Smooth Ball using fine mesh for both cases, the obtained results are compared against the experimental results found in Aoki et al., 2010.

From Fig.31 above and the values from the table 4, Smooth Ball has upto 68% higher drag




(a)




(b)

Figure 31: (a) Plot of Drag Coefficient for varying RPM and $\alpha$ (b) Plot of Lift Coefficient for varying RPM and $\alpha$

| RPM | Alpha | Golf Ball | | Smooth Ball | | Error | CD Comparison |
|---|---|---|---|---|---|---|---|
| | | CD | | CD | | | |
| | | Experiment | Simulation | Experiment | Simulation | | |
| 2000 | 0.1 | 0.25 | 0.24 | 0.42 | 0.4 | 4% | 68% |
| 3000 | 0.15 | 0.27 | 0.265 | 0.41 | 0.39 | 4% | 51% |
| 4000 | 0.2 | 0.3 | 0.29 | 0.402 | 0.38 | 5% | 34% |

Table 4: Golf Ball Vs Smooth Ball Drag Coefficient Comparison with experimental data (experiment found in Aoki et al., 2010)

coefficient than Golf Ball. As expected Smooth Ball has higher drag coefficient than Golf

Ball, because the wake generated by Smooth Ball is larger than the wake generated by Golf Ball. Smooth Ball has uniformly smooth surface compared to Golf Ball with uneven surface uniformity due to the presence of dimples, this uneven surface in Golf Ball causes the flow to separate earlier than Smooth Ball, as it creates small turbulent regions near the dimples as the surface is uneven. This early flow separation helps the flow to reunite earlier than Smooth Ball. Having the flow reunite earlier leads to smaller wake region formed behind Golf Ball, smaller the wake region, smaller the turbulence flow inside the wake region. Hence, the drag force acting on the Golf Ball is minimum in comparison to the drag force acting on the Smooth Ball. Therefore, lower Drag Coefficient is obtained for Golf Ball than Smooth Ball.

| RPM | Alpha | Golf Ball | | | | | |
|---|---|---|---|---|---|---|---|
| | | CL | | Error | CD | | Error |
| | | Experimental | Simulation | | Experimental | Simulation | |
| 2000 | 0.1 | 0.11 | 0.1 | 9% | 0.25 | 0.24 | 4% |
| 2500 | 0.125 | 0.12 | 0.112 | 6% | 0.26 | 0.25 | 3% |
| 3000 | 0.15 | 0.15 | 0.135 | 9% | 0.27 | 0.265 | 2% |
| 3500 | 0.175 | 0.2 | 0.182 | 9% | 0.28 | 0.27 | 3% |
| 4000 | 0.2 | 0.18 | 0.174 | 3% | 0.3 | 0.29 | 2% |

Table 5: Error in Lift and Drag Coefficients between Experimental and Simulation Results (experiment found in Aoki et al., 2010)

As the RPM increases, the drag coefficient of Golf Ball can be seen increasing, this is because as the Golf Ball rotates faster the uneven surface tends to act like smooth surface due to the speed of rotation. As the Golf Ball rotates faster and faster it can be seen approaching the Drag Coefficients of Smooth Ball. This implies that as the Golf Ball rotates faster the effect of dimples on Golf Ball in flow separation isn't effective enough, this maybe overcome by changing the size of the dimples or by changing the shape of the dimple.

On studying the Lift Coefficient Plot, we can notice that the Lift Coefficient of the Golf Ball tends to fall after 3,500 RPM, this can be identified as the Stall region. As the Golf Ball's uneven surface is unable to separate the flow earlier as it spins faster, it tends to act like uniform surface and more flow separation takes places causing the ball to generate less Lift. However, this fall in Lift Coefficient can also be noticed on the experimental data. Hence, the simulation has captured the stall of Golf Ball accurately to experimental data.

The above Fig.32 is plotted for Drag Coefficient obtained using two different meshes and by fixing K-Omega as the viscous solver, coarse and fine mesh and are compared against the experimental data. As expected the results obtained using fine mesh is closer to experimental data than the results obtained using coarse mesh. Fine mesh is able to obtain much accurate results as there are more number of elements in the mesh domain, resulting in less truncation error when computing between nodes. Whereas the results obtained using coarse mesh has nodes far from each other, the truncation error increases.

The above Fig.33 is plotted for Lift and Drag Coefficients obtained using various viscous models, which are, K-Epsilon, K-Omega, Transition SST, Transition K-Kl Omega and Reynolds Stress. This study is performed by fixing Golf ball spin at 4000 RPM and $/alpha$ at 0.2 $\circ$. K-Omega and Reynolds Stress viscous model was able to obtain the most accurate value for Lift and Drag Coefficient. On Varying the viscous model, it was

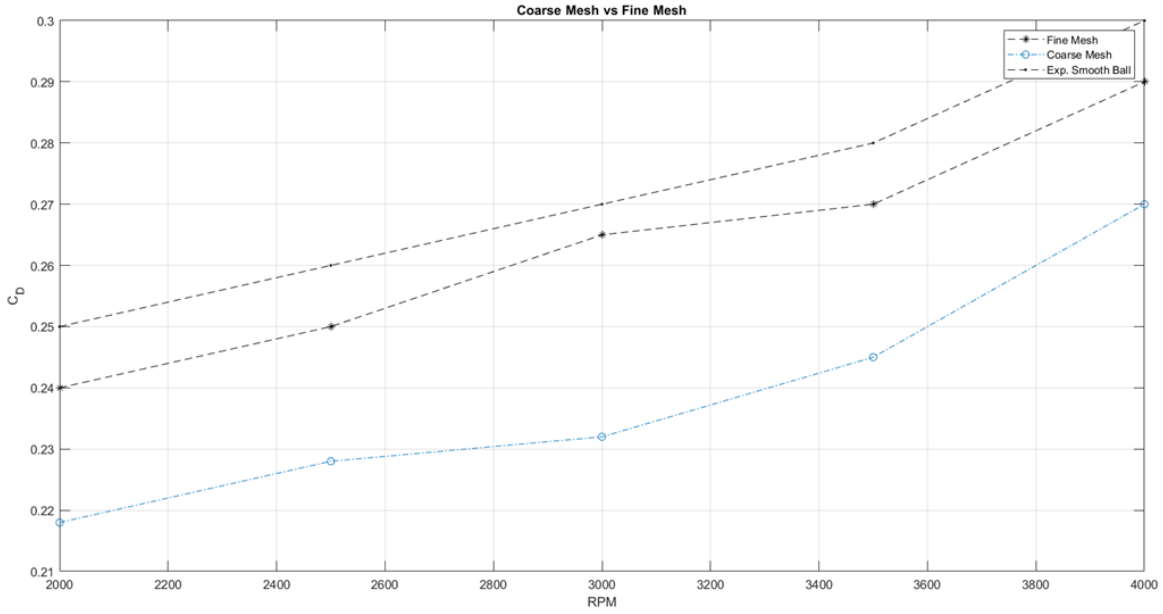



Figure 32: Plot of Drag Coefficient Obtained using Coarse and Fine Mesh

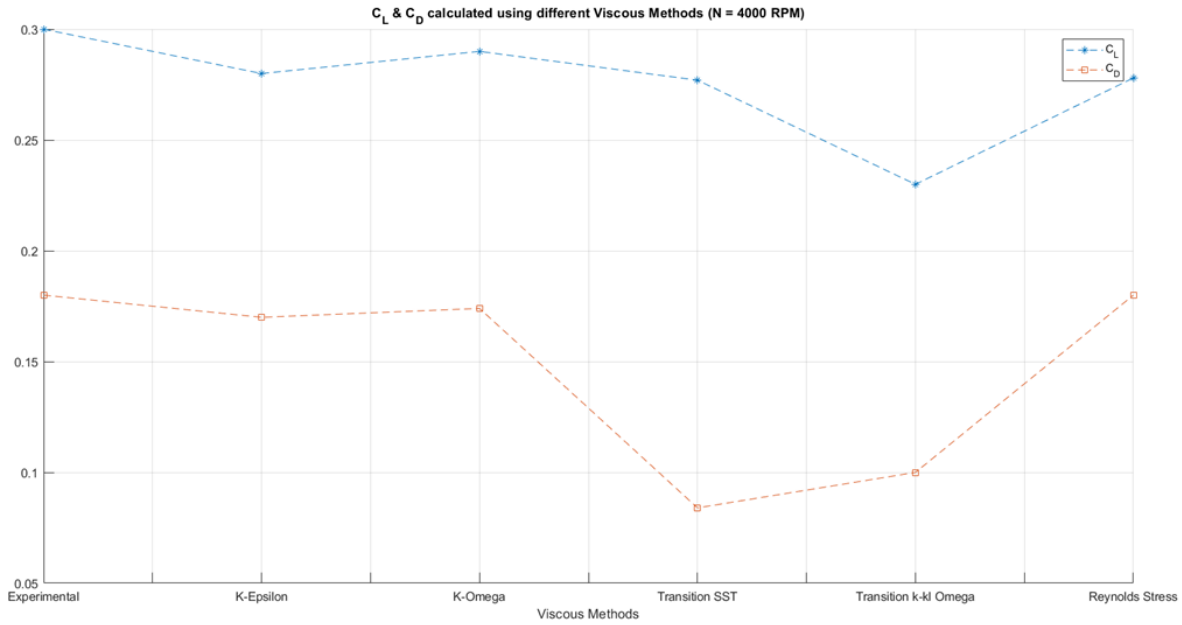



Figure 33: Study of different Viscous Models

observed that computation time increased upon going from K-Epsilon to Reynolds Stress Model, as these models become computational heavy. K-Omega model was able to obtain the results quicker than Reynolds Stress model, because Reynolds Stress Model has more equations to solve for than K-Omega model. K-Omega model has to solve for only two equations namely turbulence kinetic energy (k) equation and () is the specific rate of dissipation equation whereas Reynolds Stress Model has more equations to solve for.

In the above study, Fig.32, Fig.31, K-Omega model was used to study the results instead of Reynolds Stress model to save time. Since, K-Omega is able to obtain close results to experimental data, and is also able to obtain the results quicker than Reynolds Stress Model. But, K-Omega is computationally less heavy than the Reynolds Stress model. the above study was conducted by fixing K-Omega as viscous model.

## 4.2 SE

Here are the result from the applied methodology in section 3.2. We have an Sign up feature that is linked to our database, user can create account by inputting his information such as email, password and username. A message of success will be printed out whenever an account is created and stored .

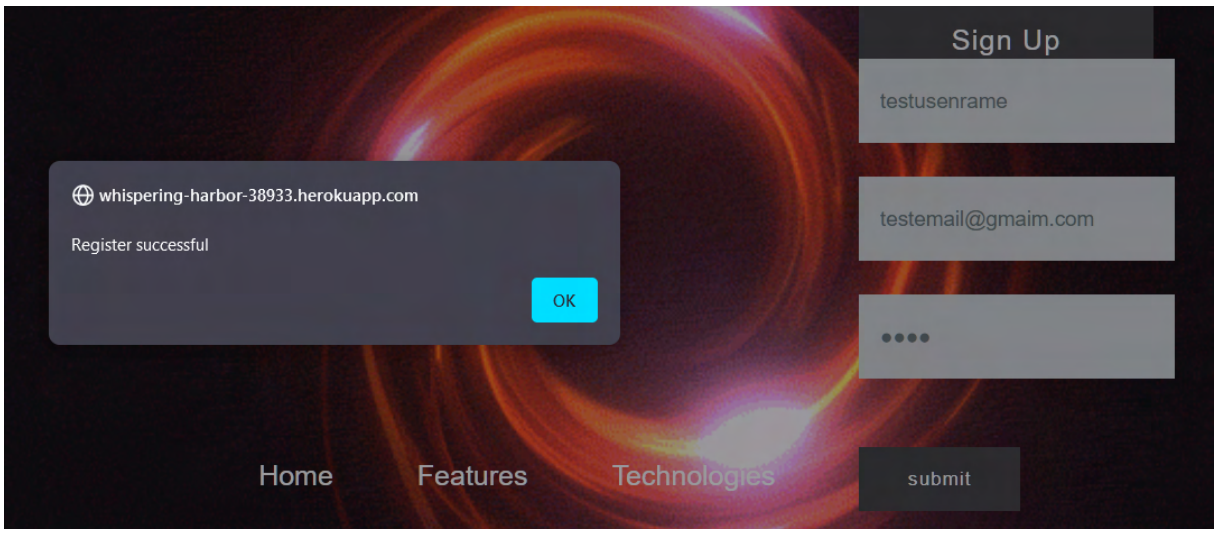



Figure 34: Successful register alert

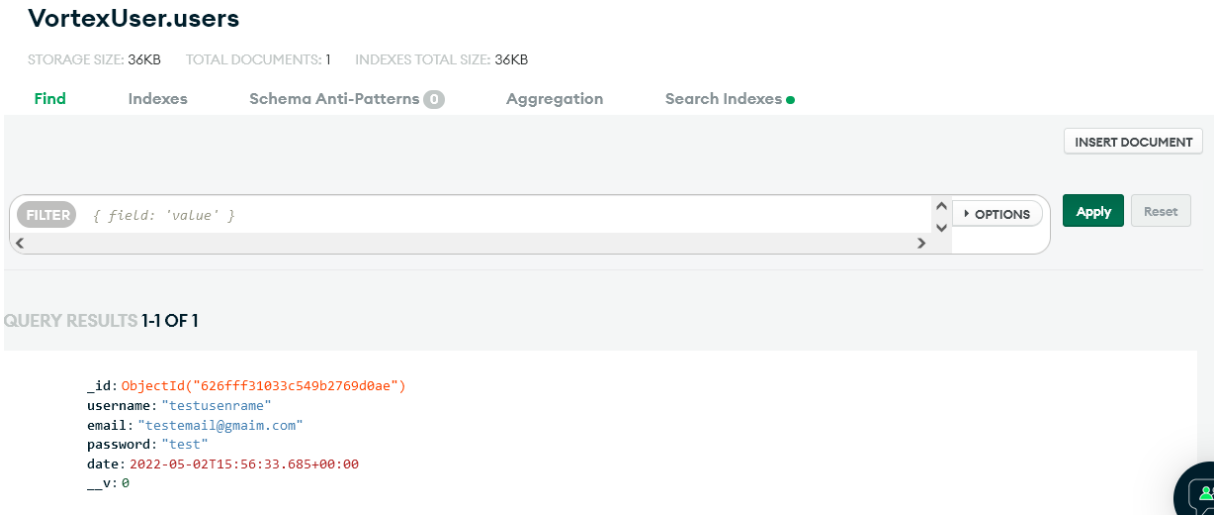



Figure 35: Added view in user database

Something similar for the authentication feature but instead of creating or adding a new user we will instead verify within our database if the corresponding information are existing or not. There's two possible outcome either it doesn't exist then a pop-up message will invite the user to verify his input, or if a valid user is found then he will be redirected to the home page .

The main feature afterwards is the detection one, the user is invited to input a picture of a vortex in the goal of findings their locations as well as their cores .

You can see aswell some default picture icon as these are the sport where the output of the machine learning code is returning to us ( or rather the data we fetch from their Apis)

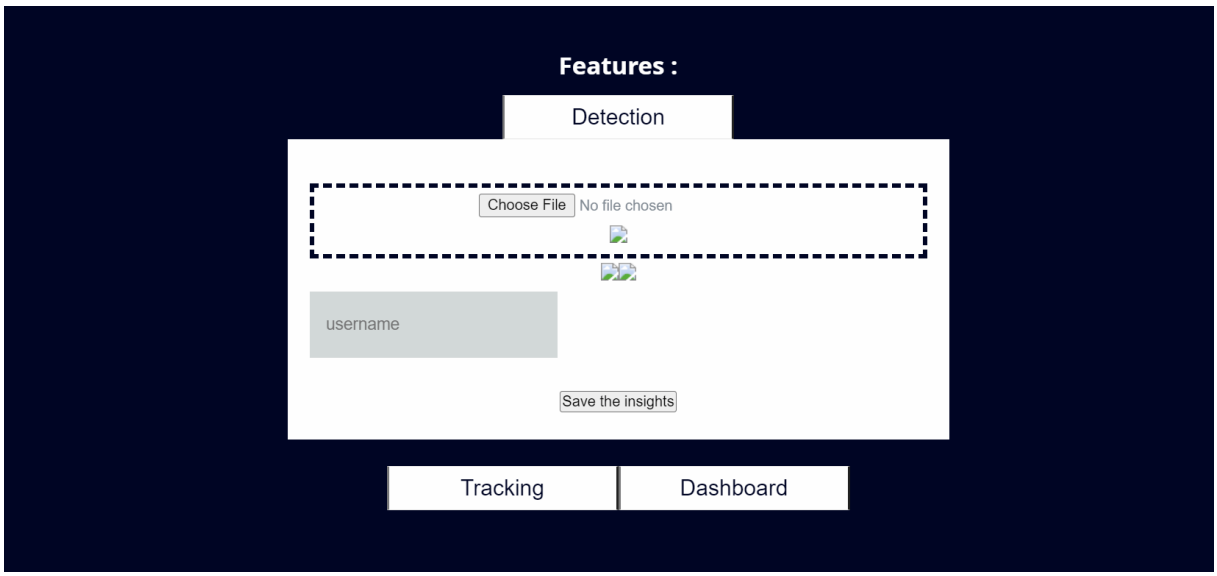



Figure 36: Detection input

first is the edited picture from the previous said input and second is a zoomed in preview of the cores locations and finaly, below will be where some information such as the x and y coordinate of the cores will also be rendered out. These output information can be also saved in the database, in order to do so, the user has to input his username to associate this feature utilisation to his name for recognition purpose .

The Dashboard whose purpose is to show every feature utilisation cases, their insight and the username associate to it as mentioned before, is yet to be rendered but we can successfully see the information passing through from the database to the application.


Figure 37: Detection Output

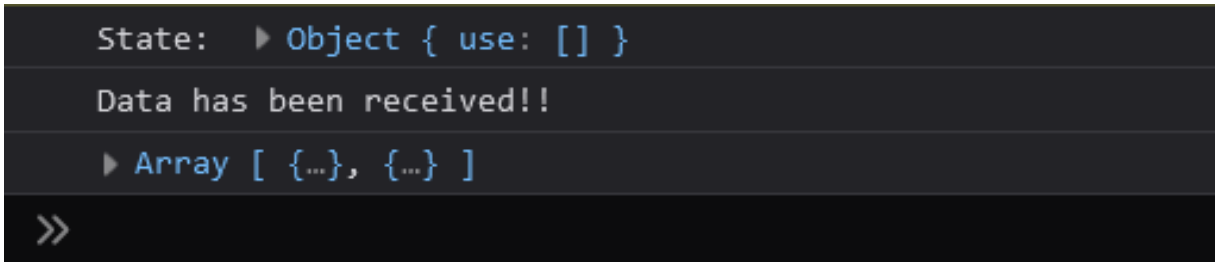



Figure 38: Data received from database to Dashboard

## 4.3 CIDA

This section explains the results from the methods and ideas explained in section 3.3. As explained in that section, we implemented several comparative tests. It's in order to find an object detection model suitable for task, the tests were conducted in 4 steps (Figure). The testing of Faster R-CNN and all YOLO models was implemented in Google Colab environment. Google Colab's environment consists of Tesla T4 GPU 16GM Memory and features 2560 shading units(cores).

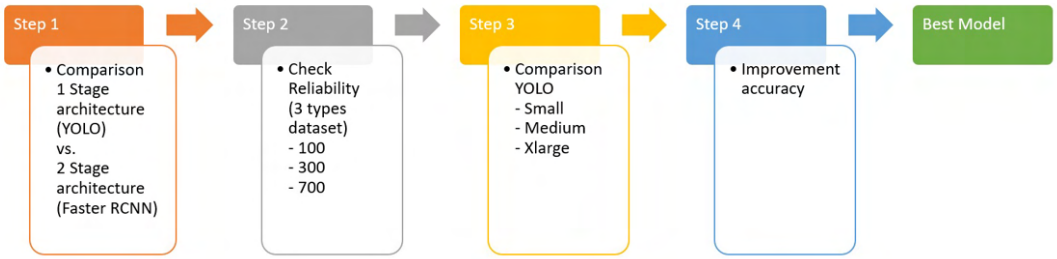



Figure 39: Test Steps for Model Selection

### 4.3.1 (Step 1) Comparison Test between YOLO v5s vs. Faster R-CNN

The first is a comparative test between 1-Stage architecture and 2-Stage architecture models. YOLO and Faster R-CNN models, which are representative models of each architecture, were selected and tested. The backbone of the YOLO model is CSPNet. The input resolution was set to 640x640. The iteration was set to stop if there was no improvement after 100 iterations, and training stopped at a sufficiently low Loss value (Box loss 0.04 or less). Resnet101 v1 was used as the backbone of the Faster R-CNN model, and the input resolution was set to 640x640. Iteration repeated learning to obtain a sufficiently low loss value (about 0.05 or less). Figure 3 is a graph of the loss value according to the training of the two models.

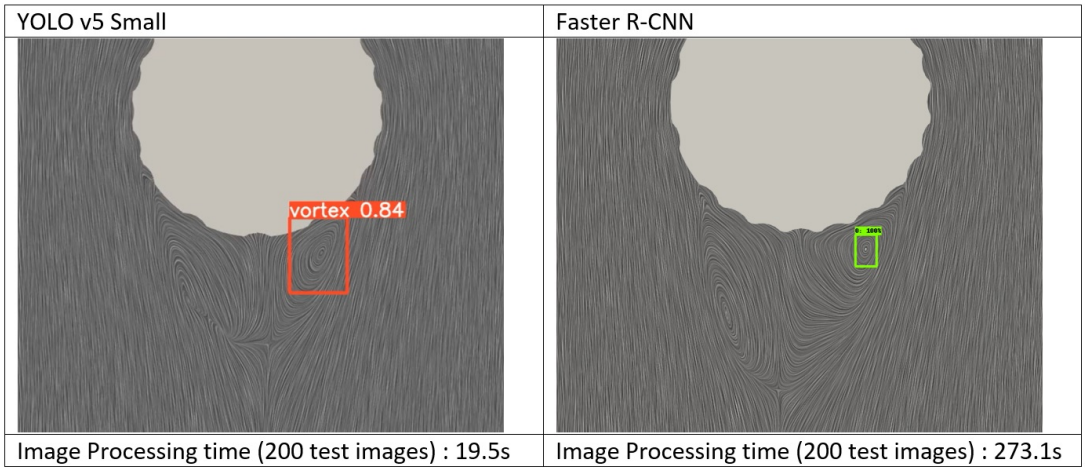



Figure 40: Vortex Detection Test Results by YOLO and Faster R-CNN model

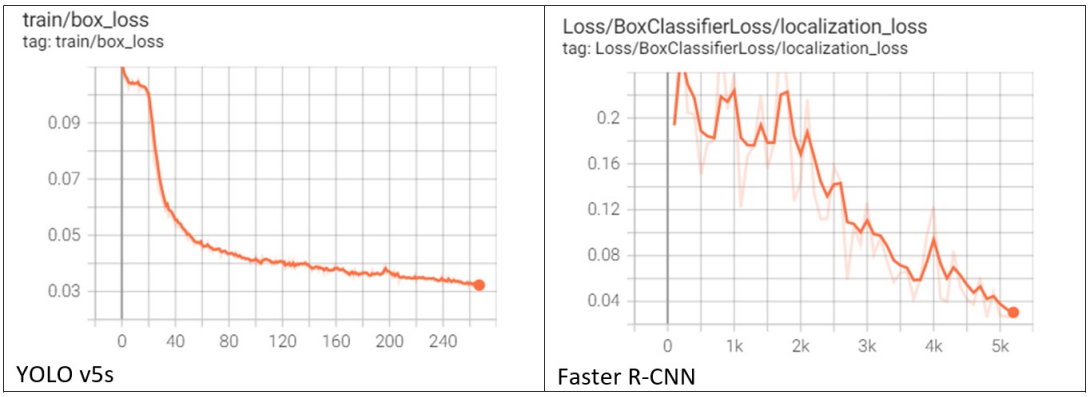



Figure 41: Training Loss Value graph of YOLO and Faster R-CNN

The purpose of the Phase 1 test is to compare image processing times. At the validation test of 200 test images, it clearly showed slightly better performance in terms of reliability, but in terms of speed, YOLO v5s took about 20s, and Faster R-CNN model took about 300 seconds. Thus, there was a large speed difference of more than 10 times.

### 4.3.2 (Step 2) Comparison Test to verify model reliability (3 types dataset)

The purpose of the Phase 2 test is to verify the reliability of the model. Overfit occurs when the training error is less than the error on the test dataset. In general, this is a phenomenon that occurs when learning with a small amount of data. In most deep learning models, increasing the amount of training dataset is the direction to implement a reliable model, and it does not adversely affect model training. In step 2, the model was trained using three different sample sizes and the results were checked. When 100 training samples were used, the mAP value was 0.84, which was not low value. However, when 300 training samples were used, the mAP value decreased. This is a phenomenon that may appear when validation data increases. To see what this means, we tested again with 700 samples. It was confirmed that mAP showed higher results than when 100 training samples were used. The experimentally derived results were able to predict the reliability problem (overfit) of the model using 100 samples. The detection determined as an actual vortex is shown in Table 1, but the result of the case using 100 training samples was not reliable.

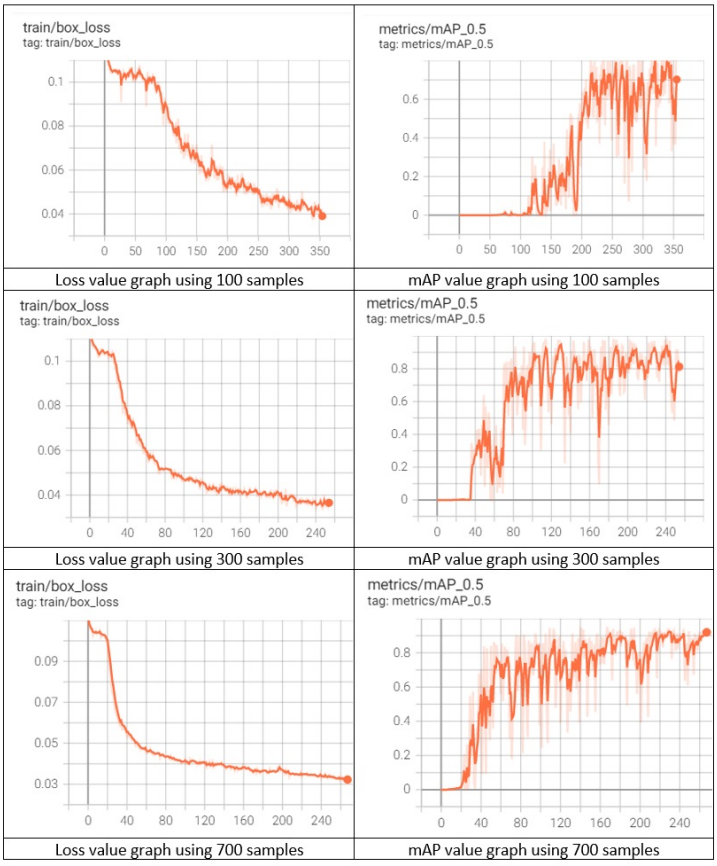



Figure 42: Training Loss and mAP Value graph by No. of Training Samples (YOLO v5s)

| | 100 samples | 300 samples | 700 samples |
|---|---|---|---|
| mAP | 0.86 | 0.72 | 0.92 |
| No. of Detection | 413 | 290 | 352 |

Table 6: Comparison of mAP and No. of Detection by No. of Training Samples

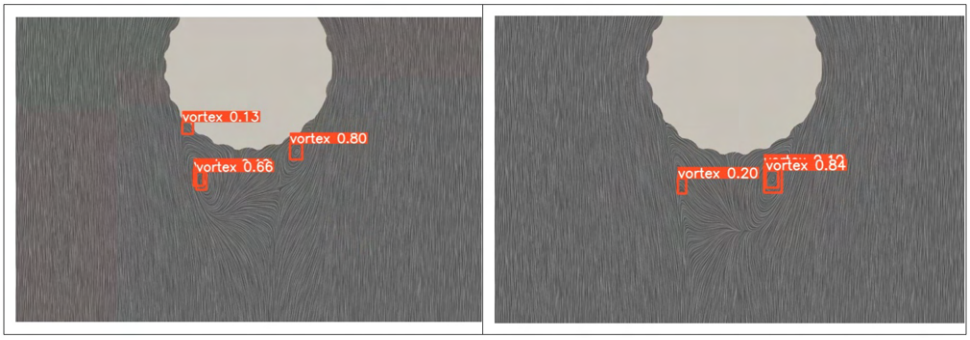
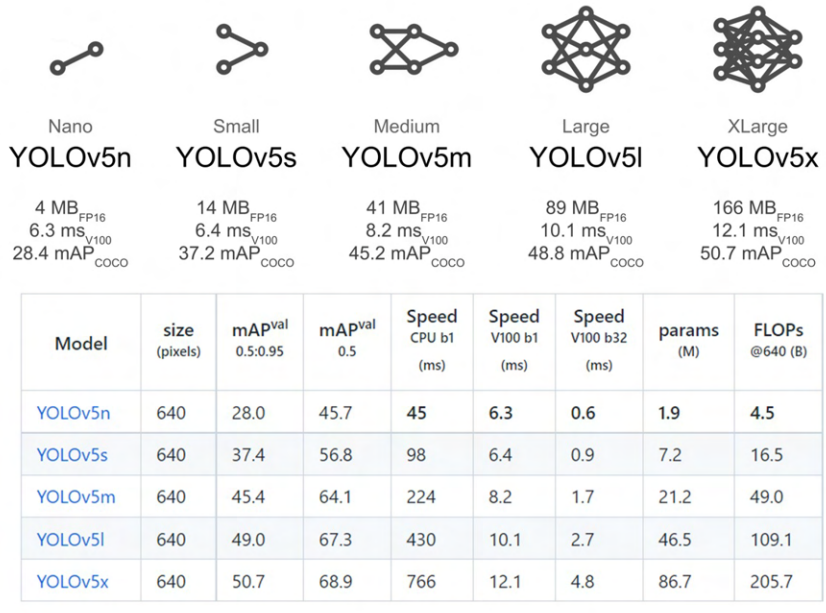



Figure 43: Vortex Detection Test Results by YOLOv5s using 100 training samples

### 4.3.3 (Step 3) Comparison Test to select suitable model

The purpose of the Phase 3 test is to select suitable model for additional learning considering cost (time consuming). In step 3, we trained the model using 3 different versions of the YOLO model and checked the results. YOLO has 5 different versions of the model. Each model has different performance. According to the information provided by YOLO, the more classes there are, the clearer the performance difference depending on the options. The selected models' performance was checked through testing after learning because our test is extraction to a custom model using only one class. The training used 700 samples. The small model showed 0.92 mAP (accuracy), Medium and XLarge models showed 0.94 mAP. 200 images were processed in 20 seconds on average for the Small and Medium models, and 25 seconds for the XLarge model.




| Model | size (pixels) | mAP val 0.5:0.95 | mAP val 0.5 | Speed CPU b1 (ms) | Speed V100 b1 (ms) | Speed V100 b32 (ms) | params (M) | FLOPs @640 (B) |
|---|---|---|---|---|---|---|---|---|
| YOLOv5n | 640 | 28.0 | 45.7 | **45** | **6.3** | **0.6** | **1.9** | **4.5** |
| YOLOv5s | 640 | 37.4 | 56.8 | 98 | 6.4 | 0.9 | 7.2 | 16.5 |
| YOLOv5m | 640 | 45.4 | 64.1 | 224 | 8.2 | 1.7 | 21.2 | 49.0 |
| YOLOv5l | 640 | 49.0 | 67.3 | 430 | 10.1 | 2.7 | 46.5 | 109.1 |
| YOLOv5x | 640 | 50.7 | 68.9 | 766 | 12.1 | 4.8 | 86.7 | 205.7 |

Figure 44: YOLO v5 Options and Performances https://pytorch.org

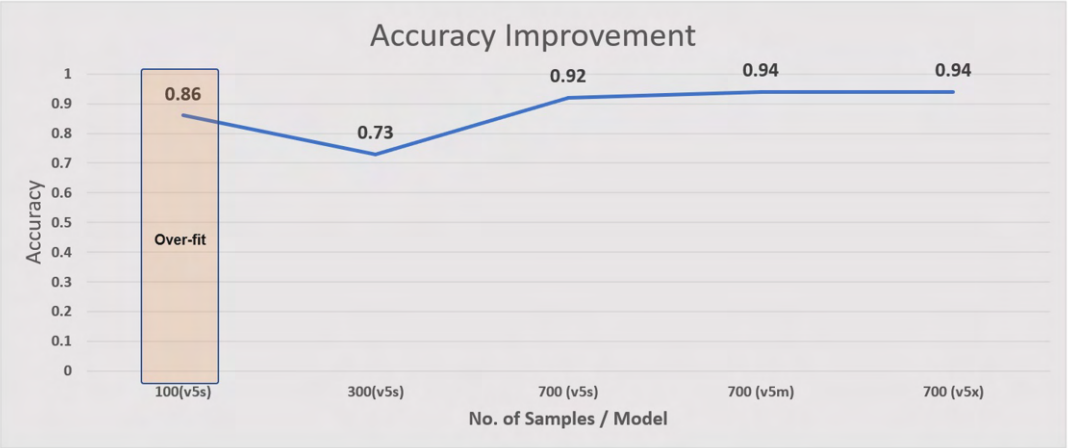



Figure 45: Accuracy Improvement of Test Model

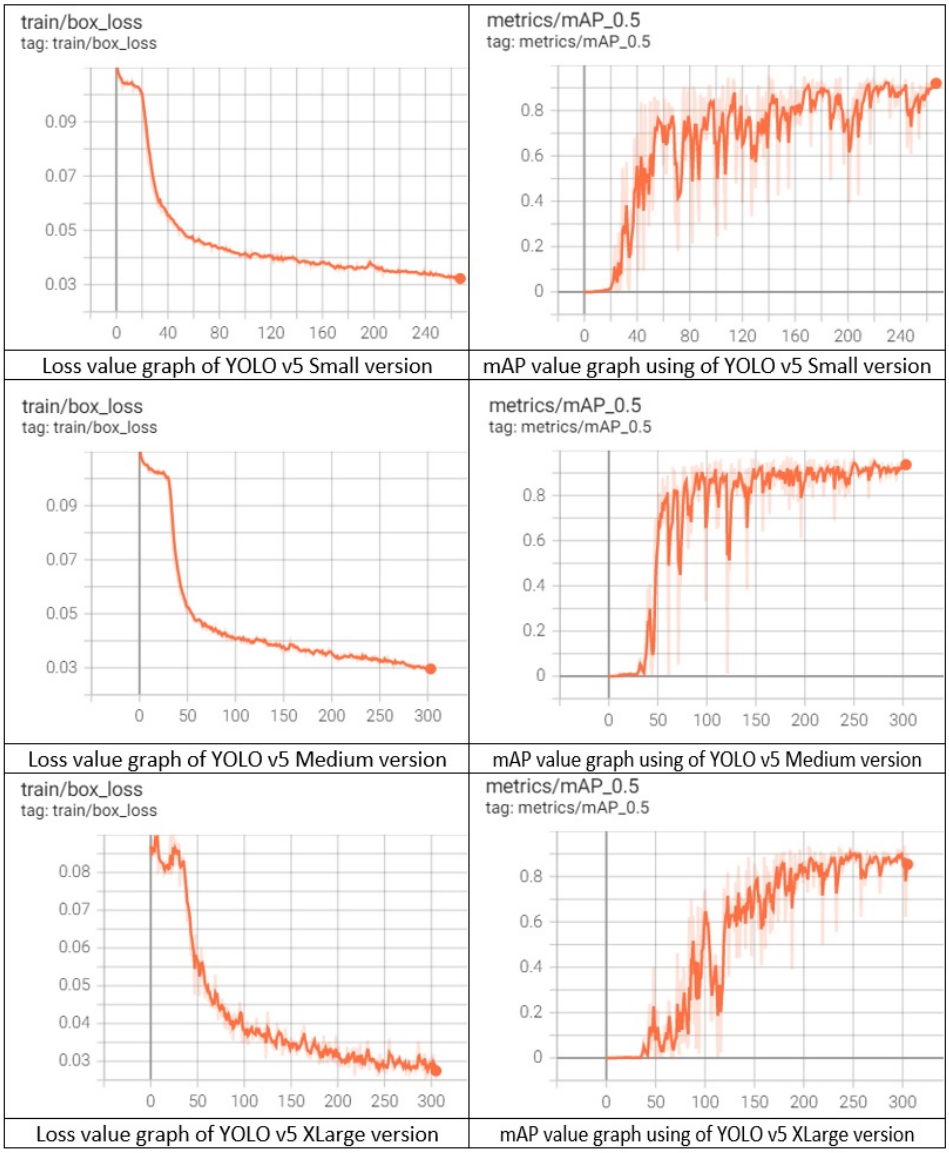



Figure 46: Training Loss and mAP Value graph of Small, Medium and XLarge options

### 4.3.4 (Step 4) Improvement Accuracy and Model Selection

Creating high-accuracy models for detecting small objects is not an easy task. Various fields are working to solve this related problem. The problem of finding the vortex can also be closer to the centre through a model with higher accuracy. The most obvious solution to this is to increase the training sample size. The recommended sample size on the official website that provides information related to YOLOv5 is 1500 or more per class (https://github.com/ultralytics/yolov5/issues/4880). Therefore, we confirmed the improvement in accuracy by increasing the size of the training samples.

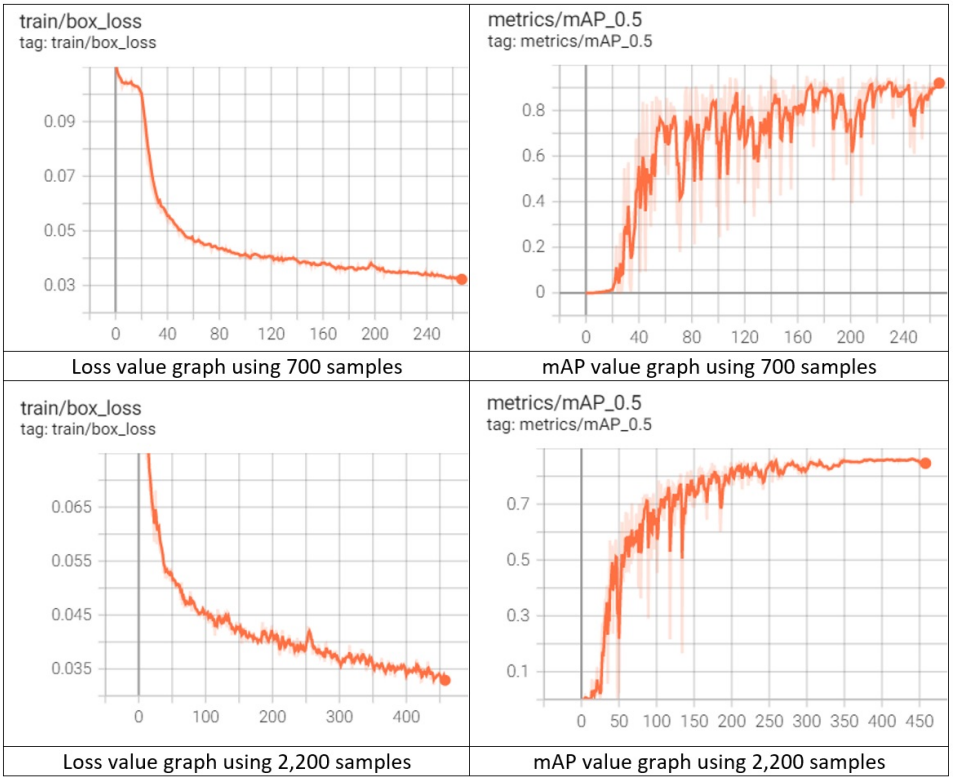



Figure 47: Training Loss and mAP Value graph by No. of Training Samples (YOLO v5m)

Indeed, the mAP value of the experimental result did not increase when 2,200 samples were applied. This is a natural result because the amount of sample for validation has increased. However, it was confirmed that the reliability confirmed in the test image increased (Figure 48).

A comparative test was performed at each step. As a result, we use the YOLO v5 Medium model trained with 2200 samples for our task. However, another method is required to access the core of Vortex. Related to this, a new approach is applied from machine vision.

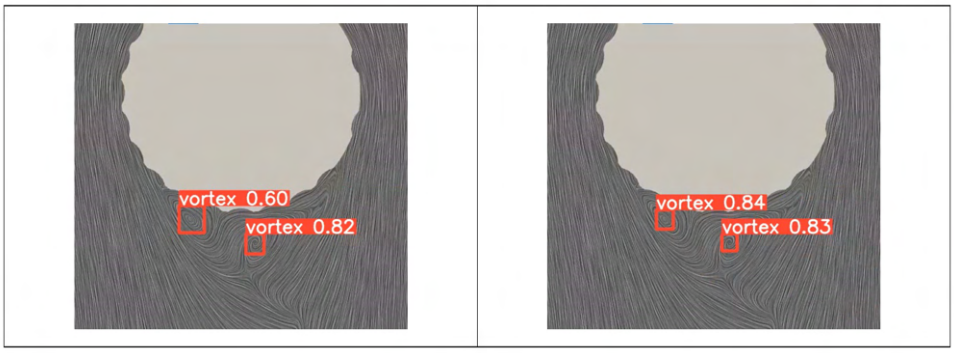
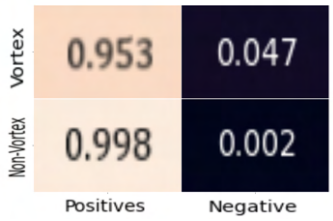



Figure 48: Vortex Detection Test Results by YOLOv5m using 700 / 2200 training samples

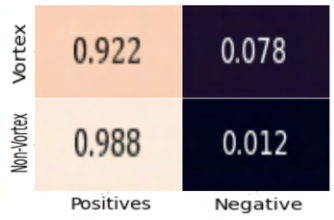
## 4.4 SIFT for vortex detection

This section displays the results from the methods and ideas explained in section 3.4. As explained in that section, we tested 4 different networks using different SIFT parameters. We trained two using $32 \times 32$ sized images and two using $64 \times 64$ sized images. For both image sizes two networks were trained using $2x2$ histogram regions and $4x4$ histogram regions to describe each box. The network itself is described in 3.4.3

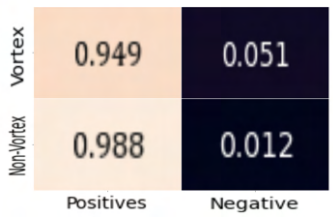

(a) $32 \times 32$ Image size, $2x2$ histogram boxes




(b) $32 \times 32$ Image size, $4x4$ histogram boxes




(c) $64 \times 64$ Image size, $2x2$ histogram boxes

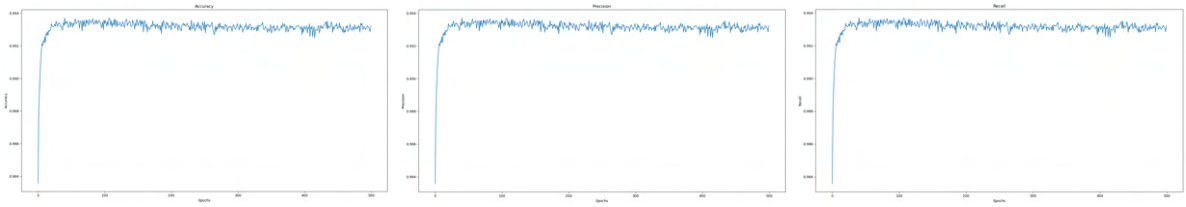



(d) $64 \times 64$ Image size, $4x4$ histogram boxes

Figure 49: The Confusion Matrices for each SIFT model

| Image Size | Histogram Box count | Overall Accuracy | Weighted F1 Score | Vortex F1 Score | Non-Vortex F1 Score |
|---|---|---|---|---|---|
| $32 \times 32$ | $2 \times 2$ | 0.9932 | 0.9932 | 0.9148 | 0.9964 |
| $32 \times 32$ | $4 \times 4$ | 0.9962 | 0.9962 | 0.9524 | 0.9980 |
| $64 \times 64$ | $2 \times 2$ | 0.9756 | 0.9754 | 0.9337 | 0.9850 |
| $64 \times 64$ | $4 \times 4$ | 0.9812 | 0.9812 | 0.9483 | 0.9885 |

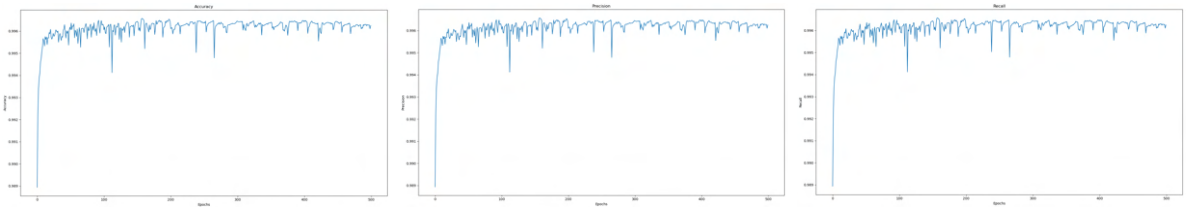
Table 7: SIFT Model Results



(a) $32 \times 32$ Image size, $2 \times 2$ histogram boxes Accuracy Graph

(b) $32 \times 32$ Image size, $2 \times 2$ histogram boxes Precision Graph

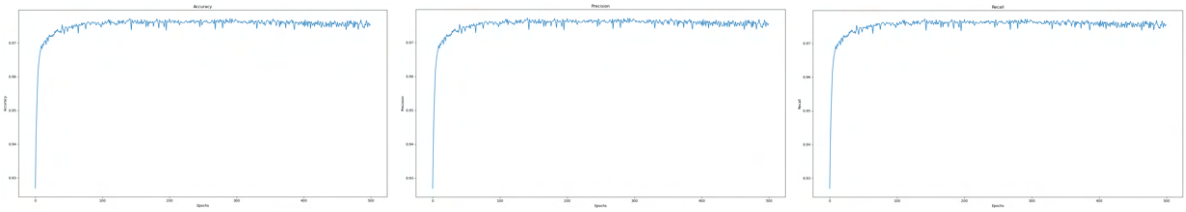
(c) $32 \times 32$ Image size, $2 \times 2$ histogram boxes Recall Graph



(d) $32 \times 32$ Image size, $4 \times 4$ histogram boxes Accuracy Graph

(e) $32 \times 32$ Image size, $4 \times 4$ histogram boxes Precision Graph

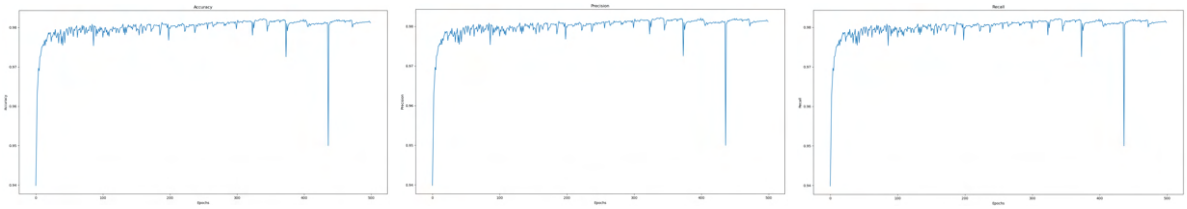
(f) $32 \times 32$ Image size, $4 \times 4$ histogram boxes Recall Graph



(g) $64 \times 64$ Image size, $2 \times 2$ histogram boxes Accuracy Graph

(h) $64 \times 64$ Image size, $2 \times 2$ histogram boxes Precision Graph

(i) $64 \times 64$ Image size, $2 \times 2$ histogram boxes Recall Graph



(j) $64 \times 64$ Image size, $4 \times 4$ histogram boxes Accuracy Graph

(k) $64 \times 64$ Image size, $4 \times 4$ histogram boxes Precision Graph

(l) $64 \times 64$ Image size, $4 \times 4$ histogram boxes Recall Graph

Figure 50: The Accuracy, Precision and Recall Graphs plot against epoch count for each SIFT model

On the surface the results seem to be reasonably effective for the classification of the images. In figure 49 we see the confusion matrices for each of the models. We can immediately in general the accuracy for labelling a non-vortex was significantly higher than a vortex. This initial difference is likely due to the difference in dataset size between the two classes, was for both image sizes we had significantly more non-vortex data images than vortex data images. In addition we see that the $32 \times 32$ sized images seemed to have a greater accuracy over the non-vortex class. This is likely because there were significantly more non-vortex images in the $32 \times 32$ dataset. This is also indicated by the fact that the weighted F1 scores in table 7 are much more heavily weighted towards the Non-vortex result in each model.

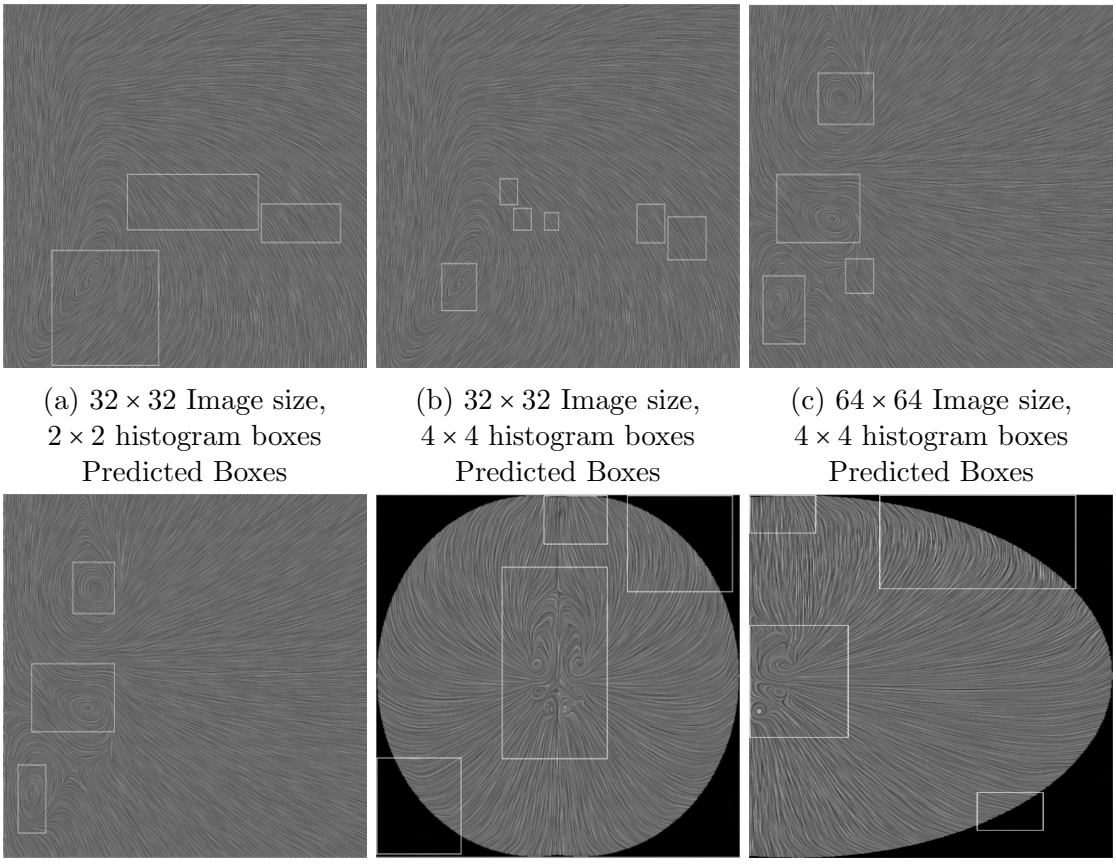
We also see that both the models which use $4 \times 4$ histogram boxes achieved a higher

classification accuracy for the vortex class than their $2 \times 2$ counterparts in both image size cases. This trend implies that with a greater number of region descriptors we achieve more accurate classification results on our dataset. This trend is also backed up by the larger Vortex F1 scores shown in table 7 for the $4 \times 4$ region descriptors, meaning that the methods also are also not over-fitting heavily and that we are not biased towards one specific class.

In addition, figure 50 displays how models behave during the training process on the validation set. For the most part, the Accuracy, Precision and Recall graphs each follow a very similar trend for each model. We see no clear signs of over-fitting during this process except for a few steep drops in a few of the graphs. In addition the curves appear to even out quite nicely and seem to settle at a specific value.

### 4.4.1 Validation

Even though we find a reasonable performance with each of the models, we do encounter issues when we try to apply any of the models to an actual image. Figure 48 displays some of the example results. To determine the predicted vortex regions within an image we apply a sliding window over the image and determine the SIFT feature at each window position. Each feature is then passed through the model and the region is labelled as a vortex if the region is above a certain probability threshold.



(a) $32 \times 32$ Image size, $2 \times 2$ histogram boxes Predicted Boxes

(b) $32 \times 32$ Image size, $4 \times 4$ histogram boxes Predicted Boxes

(c) $64 \times 64$ Image size, $4 \times 4$ histogram boxes Predicted Boxes

(d) $64 \times 64$ Image size, $2 \times 2$ histogram boxes Predicted Boxes

(e) $64 \times 64$ Image size, $2 \times 2$ histogram boxes Predicted Boxes

(f) $64 \times 64$ Image size, $2 \times 2$ histogram boxes Predicted Boxes

Figure 51: The bounding box predictions from applying a sliding window to pass SIFT features into the model.

We see the results on actual image data gives mixed results. When using a smaller image size we get quite a larger number of misclassifications and generate much larger regions than the vortices themselves. When we use the $64 \times 64$ image sizes we tended to see more accurate results on clear images with large vortices, however it begins to fail when we use smaller images with more vortices within a clustered region. Quite often neighbouring vortices would be labelled as one large region and sometimes the edges of the image itself would be detected as a vortex.

The issue of the image edges being labelled as vortices is most likely due to the strong edge responses we tend to find in those regions. In addition the neighbouring regions being classified into one singular region is more due to the method applied to generate those regions. A very naive approach is taken where we combine all the connected boxes into one singular box, meaning the final box locations is more of a prediction as to a region in which a vortex might be found.

Finally, the result of the larger boxes producing more accurate responses on these larger images, is most likely due to the reduced feature variance when using large histogram boxes. By using 4 histogram boxes on a $64 \times 64$ sized image, each histogram box covers a $32 \times 32$ region, meaning the data covers a wider variety of pixel responses. Due to this the impact of outliers is most likely minimised due to this and we tend to see less variance in outlying model responses. We do still get these misclassifications on the edges of the image, but this could potentially be removed with some pre-processing. In addition pre-processing could help reduce the impact of the aforementioned outliers and investigations should be done to explore new routes to take this concept. For the final implementation of the software, we decided to omit these models since they need more work before they can be used properly and the YOLO algorithm was used instead.

## 4.5 Core Location

This section displays the results of a variety of tests using different parameters for the algorithm described in 3.5. Table 8 and Figure 52 display the results of the tests that were ran for this algorithm for the vortex core finding. The error stated is the mean distance, across each vortex, from the manually labelled vortex core location. In addition the error 'benchmark' of randomly choosing the centre of the bounding box as the core location was used to compare the results of this algorithm to another method. We also chose this benchmark because it could be seen as the naive method for determining the core's location.

Overall we see that for the majority of tests we perform better, on average, than the previously stated benchmark. Every test, except for 3 with very extreme parameters, performed more accurately than the naive approach. In addition to testing the values of the parameters here, 4 tests were also done using the Gaussian and Median image filters which are explained by Szeliski, 2022. They were applied to the output of the voting stage to 'smooth' the resulting probability distribution such that the gaussian fitting step would ideally produce better results. By comparing tests 7-10 to test 1 we see that these smoothing operators do not produce a reduced error, but actually increase the error size. We do however slightly reduce the standard deviation, which means the variation in the outputs does appear to be reduced by smoothing the probability distribution. In addition applying a stronger smoothing operator appears to further reduce the error.

Another point to note is that even though the results are on average better than the naive

| Test Number | Sample Size | Update Rate | Iterations | Notes | Mean Error |
|---|---|---|---|---|---|
| Test 1 | 10,000 | 3 | 100 | | $1.4386 \pm 1.2104$ |
| Test 2 | 10,000 | 3 | 200 | | $1.5740 \pm 1.3177$ |
| Test 3 | 10,000 | 3 | 30 | | $2.0972 \pm 2.2244$ |
| Test 4 | 500 | 3 | 100 | | $2.1992 \pm 1.8043$ |
| Test 5 | 1000 | 3 | 100 | | $1.9814 \pm 1.6275$ |
| Test 6 | 1000 | 3 | 30 | | $2.1376 \pm 1.7717$ |
| Test 7 | 10,000 | 3 | 100 | Median Filter | $1.6695 \pm 1.2451$ |
| Test 8 | 10,000 | 3 | 100 | Gaussian - 1.0 sigma | $1.5353 \pm 1.1789$ |
| Test 9 | 10,000 | 3 | 100 | Gaussian - 1.5 sigma | $1.5366 \pm 1.1508$ |
| Test 10 | 10,000 | 3 | 100 | Gaussian - 2.0 sigma | $1.5282 \pm 1.1459$ |
| Test 11 | 100,000 | 3 | 100 | | $1.3071 \pm 1.0440$ |
| Benchmark | - | - | - | Boudning Box Centre | $2.0456 \pm 1.3330$ |

Table 8: Table of Tests for different parameter values for the Gradient Descent-like algorithm

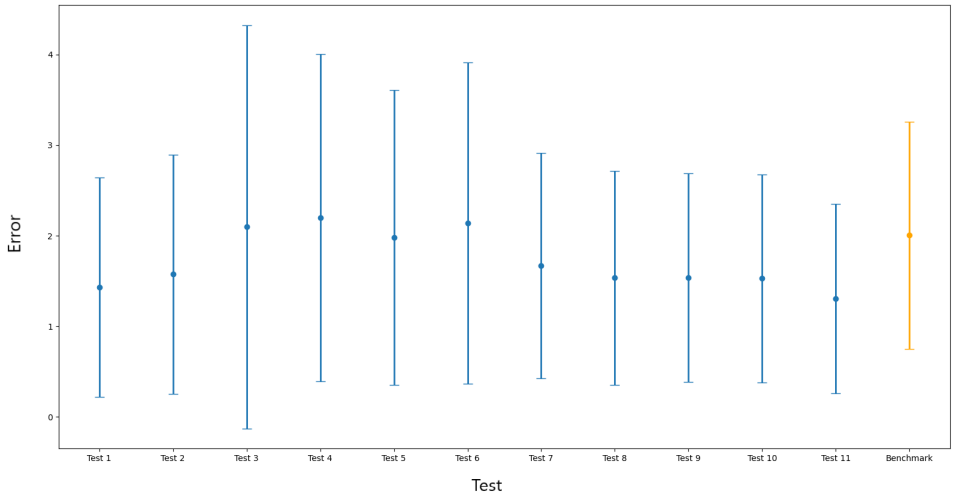


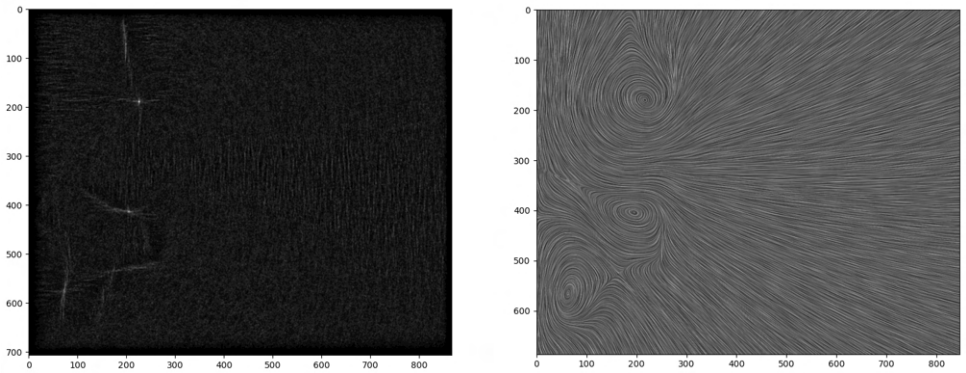

Figure 52: A graphical Display of the Results of Table 8 along with the benchmark displayed in orange

approach, we still get very large error bars. This issue is most likely due to the inherently random approach that we have to take here. There is also a trend amongst the results that these error bars are reduced as the mean error is decreased. In addition there is some randomness in the human labelling as well. For the most part the standard deviation is within 1-2 pixels. It is not entirely unreasonable to say that the expected human error might align with these values and the manual labelling may be off from the true core location by 1-2 pixels as well.

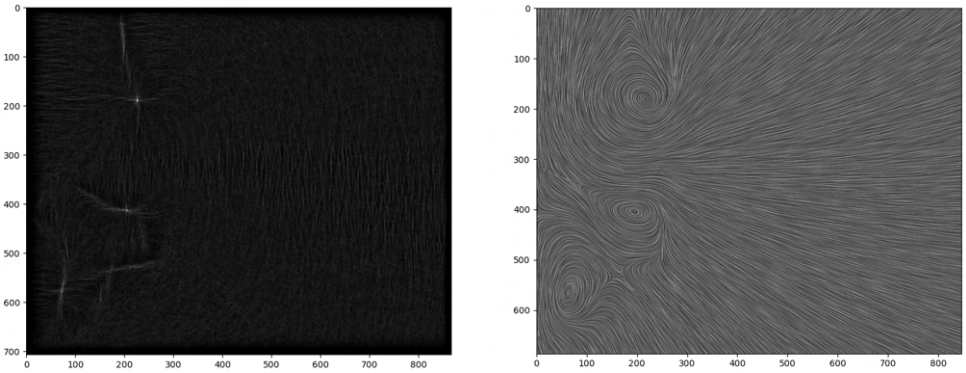
We see that the method which minimises the mean error here is Test 11, where we apply the greatest sample size. We do also see that as the sample size is increased, the mean error is reduced, with Test 11 having the greatest sample size and the smallest mean error. Even though this performs the most accurately, we opted for the parameters shown in Test 8 for the final software. Since we aimed to make an application that would work in real-time the time taken to run the program was also a major factor in this decision.

With Test 11 we use 10× the sample size as that in Test 8, which means there are 10× as many samples we need to process for each vortex in our program. We chose Test 8 over Test 1 due to the reduction in the standard deviation that the Gaussian Filter gave us. Since a randomly behaving program is not ideal for the software we were aiming for, we decided to apply the gaussian filter to help reduce the randomness of our program.

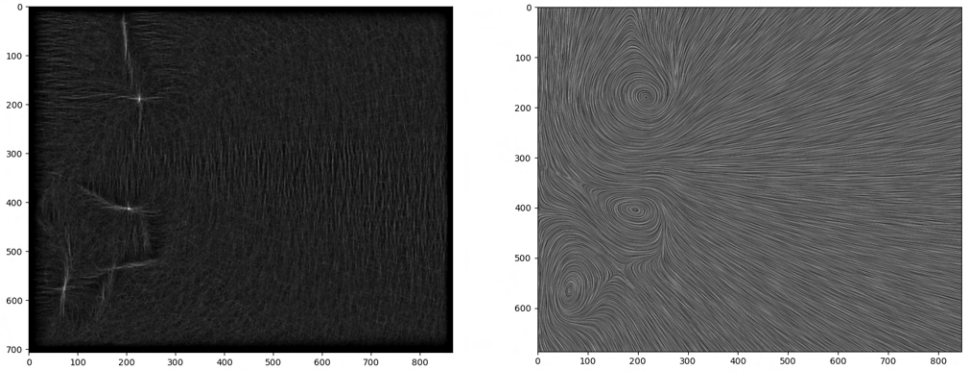
### 4.5.1 Additional Tests beyond Initial Scope



(a) Sample size of 1,000,000



(b) Sample size of 5,000,000



(c) Sample size of 10,000,000

Figure 53: The probability distribution results on a full vortex image. The sample size is varied, but the other parameters are the same as those for Test 8 in table 8

This section explains some additional behaviours of the gradient descent-like program that was developed and testing in this section. While nearing the end of the development and testing of this program we decided to test what would happen if we applied the program to the full image files, rather than a sub-image of a vortex. The results of the program are displayed in figure 53.

As you can see in the images of figure 53, we would tend to get strong responses around the locations of vortices and we would obtain interesting formations that would appear to mimic the vortex orientation and flow in the images. To perform this the sample size had to be increased to a much larger size than was used for the smaller sub-images just of the vortices as a much larger region had to be covered. We also see as the sample size is increased, we tended to see less noise within the images themselves and the vortex locations became much more distinguishable.

Even though the results are noteworthy, no proper experimentation or evaluation was done using these results as this application was outside the initial scope of the algorithm. As such this application was not included in the final software. In addition this processing would take a much larger time than using a Convolution Neural Network like YOLO.

# 5 Future Work

## 5.1 CFD

The Simulations and investigations conducted so far are fascinating and informative. However, some more interesting can studies can be done:

- Study the effect of dimple shape of the ball on the performance of the ball by varying the dimple shapes.
- Try Simulating the problem using dynamic meshing and study the effect of computational time and accuracy.
- Study the effect of a range of Reynolds Number on the performance of the Golf Ball
- Study the range and altitude of the ball for different dimple shapes
- Simulate the ball when the ball gets hit by Golf Stick Club and Study the vortex generated when the golf ball undergoes deformation when hit and when the ball gets back to its shape and continues its trajectory.

## 5.2 SE

Our web application allows user to create their own account with their personal information, however after a successful login with the authentication feature, the user is then redirected on the home page but has no evidence of being correctly logged in. Currently we have something that will take the input information and compare it to the data from mongodb then will show a message alert, depending on its success,to have something cleaner we could store the logged-in data to a local variable and print the username in the home page afterwards. This feature is only useful if there is more accessibility for an authenticated user .More so, we would like to improve on containerizing the application and configuring a working build pipeline for application build and deployment.

Another improvement, we wanted to finish implementing was the tracking feature, it was finished by the AI team but not implemented yet, the idea is similar to the detection one, the user input a video, then goes through the machine learning algorithm then output-ed on the web interface. So instead of a converted base 64 image, we should have a .mp3 type or alike types .

The last feature that need improvement is the dashboard that is supposed to have the utilisations informations such as insight ,date of use etc. Currently all the informations from the database is correctly sended to the application but yet to be rendered. The data are stored in arrays in order for each utilisation and needs to be arranged in a more appealing way .

## 5.3 CIDA

When implementing a custom model in the deep learning field related to object detection, what is always pointed out as a limitation is whether sufficient reliable training data is secured. In this study, reliable and sufficient learning data were secured through active support of CFD and machine vision parts. However, there are not many situations like this in the field, and the task of creating a dataset itself means an increase in cost. Therefore, we need to find other alternative methods. We can consider how to create an image through a technology called GAN (Generative Adversarial Network) and use it as a dataset. Unlike neural network models that usually focus on one, GAN has two types of neural networks (generators and discriminators), and is a deep learning algorithm that aims to create fakes that look like real ones. When there is a small number of training images or in order to reduce cost, a method of increasing the training data through GAN and proceeding with learning can be attempted.

Other development to this project could be real time vortex detection which is possible using the current technologies used. As the models being used are extremely fast, it can support real time detections. As the development on RCNN models takes place, there is hope for an RCNN model that is faster than the current version and is as fast as the single stage models, this could improve the speed and accuracy drastically.

Another major improvement is a feedback loop that could be added to the complete application (similar to reinforcement learning), this would train the model on the new images that are uploaded by users. This could keep improving the model over time. If added, this loop could be vital in generalizing the model to any type of vortex image/video that is sent to the model.

## 5.4 Computer and Machine Vision

Within this report many concepts founded in Computer Vision we explored as to their practical application within vortex detection. Mixed results were achieved using such techniques and concepts, however the results shown in sections 4.4 and 4.5 still leave a lot to be explored and developed. Some such ideas and routes to take are listed here:

- The SIFT descriptor model for vortex detection still has much room for improvement. Experimentation using more classical features used in image classification could be used in conjunction with SIFT features, to more accurately classify vortex regions. In addition the outputs of models using features of different sizes could be combined into a singular network to improve the overall performance.

- Rather than using SIFT descriptors in a seperate network, the directional edge information produced using the Sobel operators could be used as the input to a Convolution Neural Network such as YOLO to more accurately determine vortex regions.

- Even though Algorithm 1 in appendix I has been shown to perform more accurately than naively taking the centre of the bounding box, there is still room for improvement in terms of the behaviour of the program itself. Currently the algorithm is inherently random and large error bars were produced when evaluating the method.

Steps could be taken to explore a route which introduces less randomness and to improve the algorithm's performance overall.

- Tests shown in section 4.5.1 produced interesting results that were not explored within this project since they were discovered late into the project and were therefore not properly evaluated and explored. Steps could be taken to explore a route using the method described there to extract information about the vortices from the probability distributions produced there. In addition a route could be explored which uses the outputs shown there in a CNN such as YOLO to experiment as to what sort of information we could produce about the vortices from the outputs shown there. Proper evaluation should also be done to back up any claims made about the algorithm's use here.

- Finally, the results displayed here that used Image data could be integrated with the pre-established methods for vortex detection using additional information generated in the simulation. Importantly, using a combination of the two could lead to a much more accurate vortex detection algorithm.

# 6 Conclusions

Overall the project can be deemed a major success. Varying levels of success were achieved according the goals outlined in the initial market analysis.

First of all, the results of the simulations using the Golf Ball case study showed significant results. The golf ball mesh was shown to have a significantly reduced drag coefficient than that which was shown for the smooth ball. In addition, investigations into the effect of RPM showed that the golf ball tended to behave more like the smooth ball as the RPM was increased. In addition the simulation results were proven to be accurate since the results we shown to be close to the simulation results.

Secondly, we achieved great success during the course of the development of our YOLO model for vortex detection. Even though R-CNN was shown to provide more precise results than our YOLO model, YOLO was chosen due to it's greater processing time. In addition multiple stages of development were performed to increase the accuracy of our model, achieving a mean average precision of 92% with the data provided.

For the investigation of the use of Computer Vision techniques into the detection of vortices and vortex cores mixed results were achieved. The SIFT model for vortex detection initially seemed promising on its own, however was deemed to be inaccurate when applied to real image data. On the other hand, notable results were achieved by applying the notion of Gradient Descent using the Sobel edge detector for core location. Our algorithm performed better on average by a distance of roughly 0.5 pixels than naively choosing the centre of the bounding box as our core location. Both achievements were especially interesting for the potential of future applications in the development of our software. The notion of Gradient Descent was also shown to have a potential interesting application in the processing of full-sized images. Also, the SIFT features provided a good classification accuracy purely on their own and additional features could be explored in conjunction with them to potentially achieve even better vortex location results.

The worked was organized in a way that every one was able to contribute and bring perspectives from their respective fields. The final result was a web application, capable of locating the vortex and core locations within a given image. Even though the application was uncompleted, there is still room for much improvement as it provides a great basis for an application with a program format which would be simple the extend.

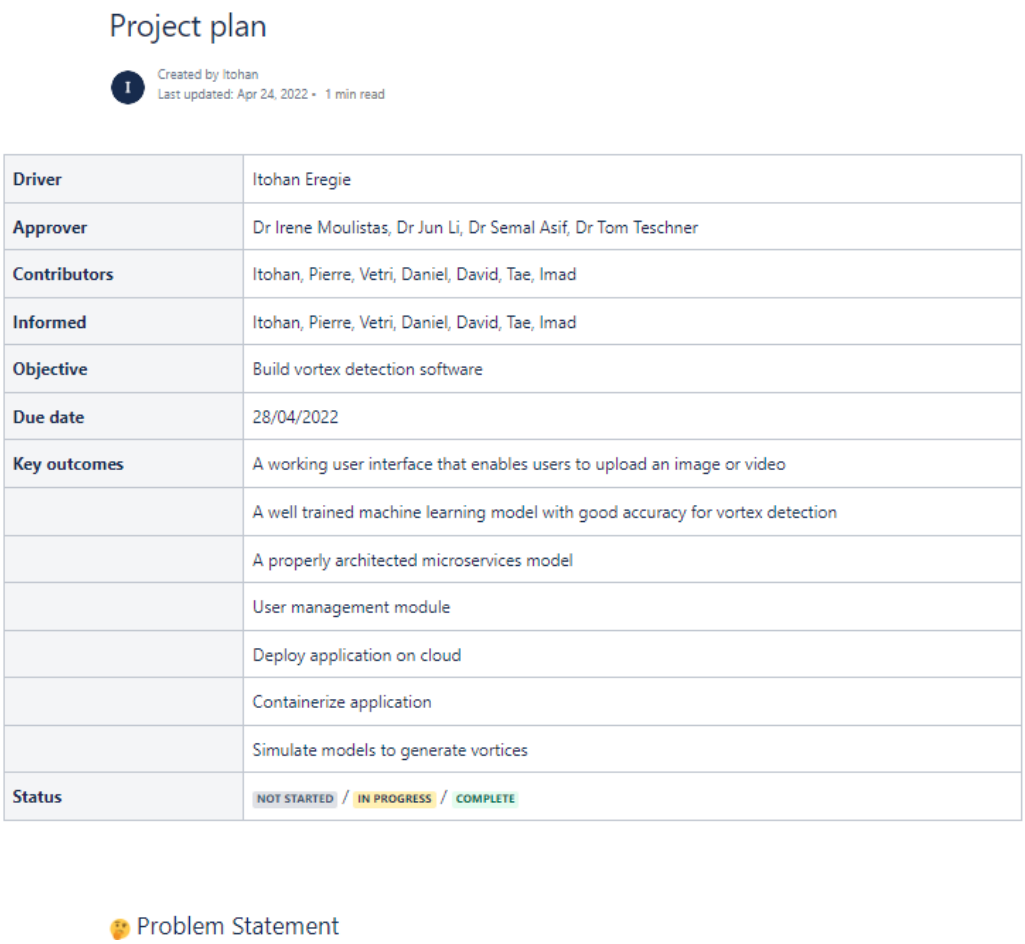
Finally, The biggest challenge we faced throughout this project was the time management. Even though we applied the agile methodology and used fixed sprint dates, research and the initial exploration stage of ideas took more time than was initially planned. The project does however open up a variety of potential routes to explore to expand on an already successful product.

# A Project Plan Documents

This is the first phase in the software development life cycle. We have used agile methodology to manage the project .Project plan document is used to outline the phases in the development process of the product.The project was split into two sprints, 1 and 2. The sprints were further broken down into smaller manageable tasks with deadlines.

## A.1 Project Plan

The scope of the project with the stakeholders, problem statement and must haves are define within this document.



Project plan

Created by Itohan
Last updated: Apr 24, 2022 • 1 min read

| | |
|---|---|
| **Driver** | Itohan Eregie |
| **Approver** | Dr Irene Moulistas, Dr Jun Li, Dr Semal Asif, Dr Tom Teschner |
| **Contributors** | Itohan, Pierre, Vetri, Daniel, David, Tae, Imad |
| **Informed** | Itohan, Pierre, Vetri, Daniel, David, Tae, Imad |
| **Objective** | Build vortex detection software |
| **Due date** | 28/04/2022 |
| **Key outcomes** | A working user interface that enables users to upload an image or video |
| | A well trained machine learning model with good accuracy for vortex detection |
| | A properly architected microservices model |
| | User management module |
| | Deploy application on cloud |
| | Containerize application |
| | Simulate models to generate vortices |
| **Status** | NOT STARTED / IN PROGRESS / COMPLETE |

Problem Statement

Figure 54: Project Plan and Scope Definition Document

### Problem Statement

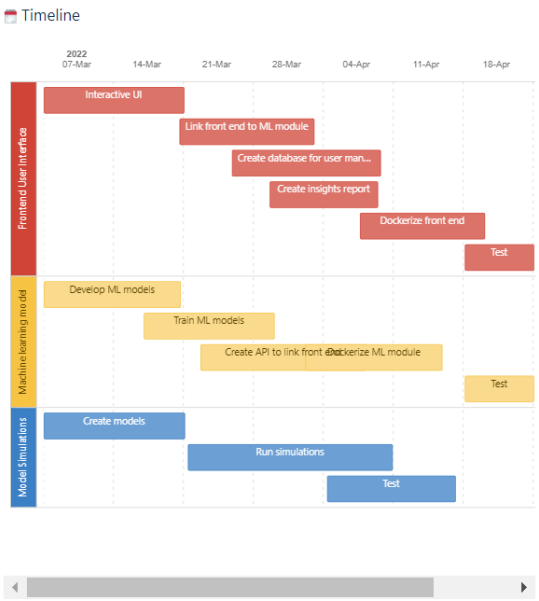
The problem statement is to detect vortices generated by objects in motion moving at a velocity . We know several machine learning models will work , but we expect the YOLO model to give us the most accurate results.

### Scope

| | |
|---|---|
| **Must have:** | • A working front end module<br>• An accurate machine learning model<br>• Images with vortices<br>• A user management module to track users activities<br>• A Database system to store simulation results<br>• A micro-services architecture<br>• A post processing module to detect vortex cores<br>• A machine learning module to locate vortices |
| **Nice to have:** | • A reports page that provides insights and compares simulations between a smooth ball and a dimpled ball<br>• Build pipeline for continuous build<br>• System monitoring and performance |
| **Not in scope:** | • <br>• |

Figure 55: Problem Scope

## A.2 Road Map




Milestones and deadlines

| Milestone | Owner | Deadline | Status |
|---|---|---|---|
| Finalize front end design | David | 25/04/2022 | done |
| Finalize AI models for object detection | Imad and Tae | 25/04/2022 | done |
| Finalize vortex simulations for mesh models | Vetri | 25/04/2022 | done |

Figure 56: Road map

The road map defines a high level pictorial view of the tasks that will be carried out during the duration of the project.Below is an image of teh road map:

## A.3 Backlog

All the tasks to be carried out are logged in the backlog. The tasks are called issues. At the beginning of the sprint, each issue is migrated to a backlog that will be monitor on a sprint board.Below is an image of the backlog:

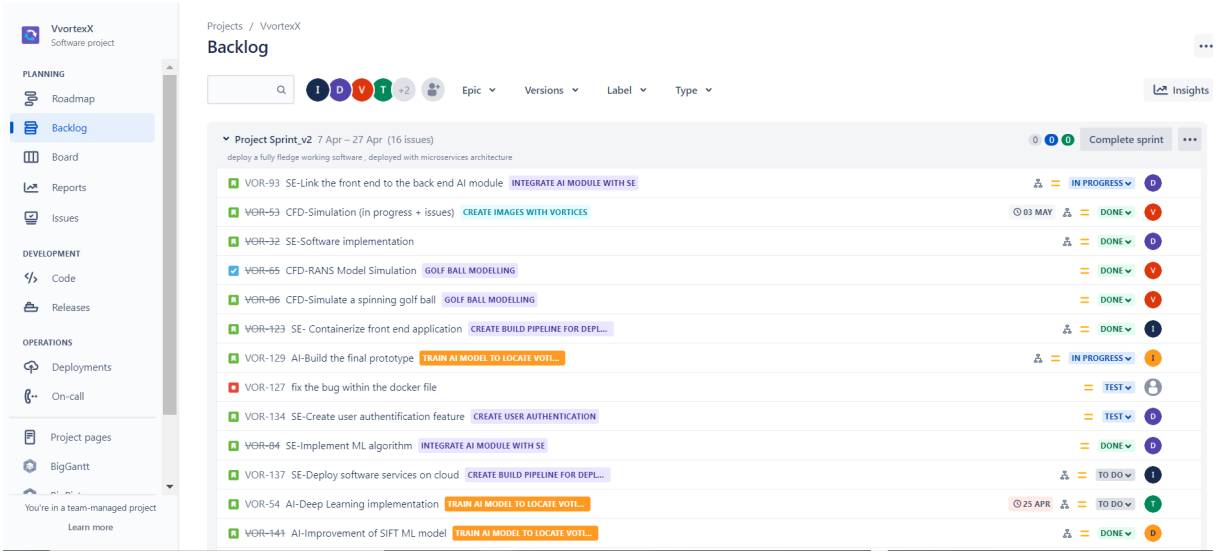



Figure 57: Backlog for Tasks

## A.4 Sprint Board

The sprint board shows a flow of the tasks in execution. It gives the current status of the tasks from TO-DO to ISSUE ONGOING to TEST and finally DONE. Below is an image of the sprint board:

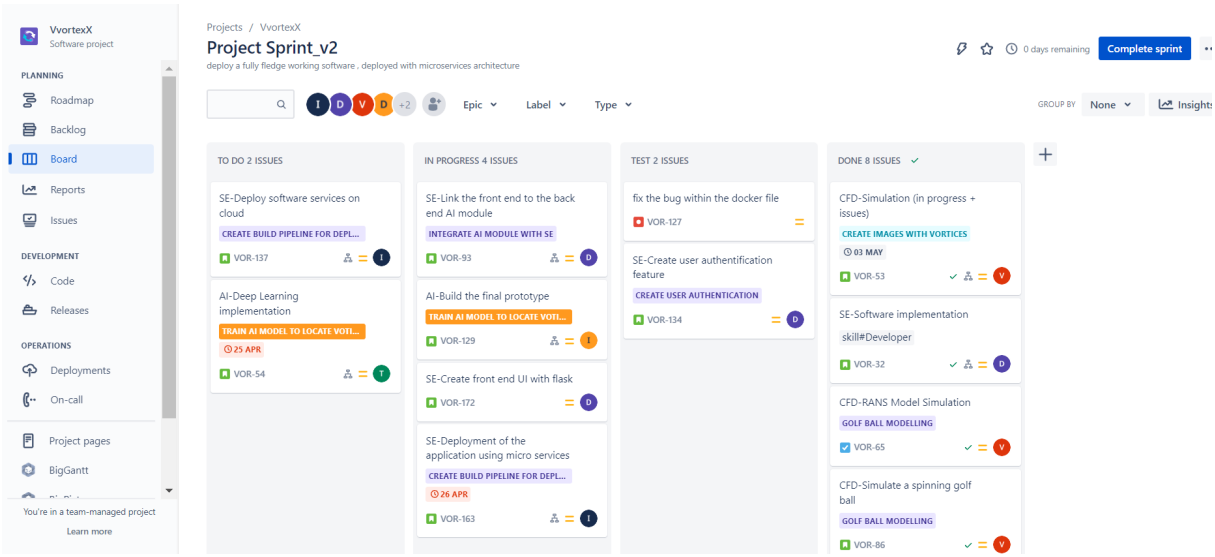



Figure 58: Sprint Board for Tasks Pipeline

# B Software Requirements and Analysis Document

## B.1 Introduction

### B.1.1 Purpose

A vortex is a region or an area in a fluid that is rotating, it is a naturally occurring phenomenon. It is formed when the fluid flows from high pressure to low pressure region resulting in a spiral shaped behavior. Vortices can move, stretch, twist and interact with the surrounding flow. The purpose of this project is to detect vortices and vortex cores produced by rotation of an object about its core. The goal of this project is developing an application to automatically locate vortices on images.

### B.1.2 Document Conventions

This document uses the conventions in Table 10.

| Convention | Meaning |
|---|---|
| SYS | An abbreviation for system |
| CICD | Continuous integration for continuous deployment |
| CMV | Computer machine vision |
| ML | Machine Learning |

Table 9: Abbreviation Conventions

### B.1.3 Intended Audience

This project has been implemented under the supervision of Dr Li, Dr Irene Moulistas, Dr. Seemal Asif and Dr. Tom Teshner. This project is useful for the manufacturing department of a golf ball company. This document is meant for software developers, machine vision engineers, machine learning engineers, computational fluid dynamics engineers, scrum masters and product owners.

### B.1.4 Product Scope

The project scope is a web-based application that detects vortices and vortex cores using machine learning. The application will have a front-end module designed as a web interface. Users will interact with the application through the user interface. The machine learning module will consist of a training model to detect vortices.

The goal of this vortex detection software is to enable design and CFD engineers automate the process of vortex location. Designers create a model to simulate the design of a golf ball, varying the design between a smooth and a dimpled ball, comparing the vortices generated by both balls.

Generating the images vortices are done outside the scope of the actual software. The simulation and design of the golf ball(s) (smooth and dimpled), are carried out within

an external system .The resulting images are uploaded to an external repository or saved locally .

The software module of this product takes in the data as an upload in image or video format, passes it to the computer machine vision module for image pre-processing. The pre-processed images are

passed through a machine learning module where the vortices will be located.

The resulting image with the vortex and vortex core locations will be returned to the font end user interface in the software module. The architecture adopted for this product is micro-services.

Micro-services architecture decouples the application making each module an independent entity. Each micro service will be packaged into a separate container or deployed in a separate virtual machine.

The project will consist of four major modules: Software engineering, Computer machine vision, Computational fluid dynamics and Machine learning.

- The software engineering module constitutes the front end and user interface of the software. Users will interact with this interface to upload data for processing in the machine learning module.
- The CMV module post processes the image, locating the vortex core.
- The machine learning module comprises of trained models that locates the accurately locates the vortices.
- The machine learning module comprises of trained models that locates the accurately locates the vortices.

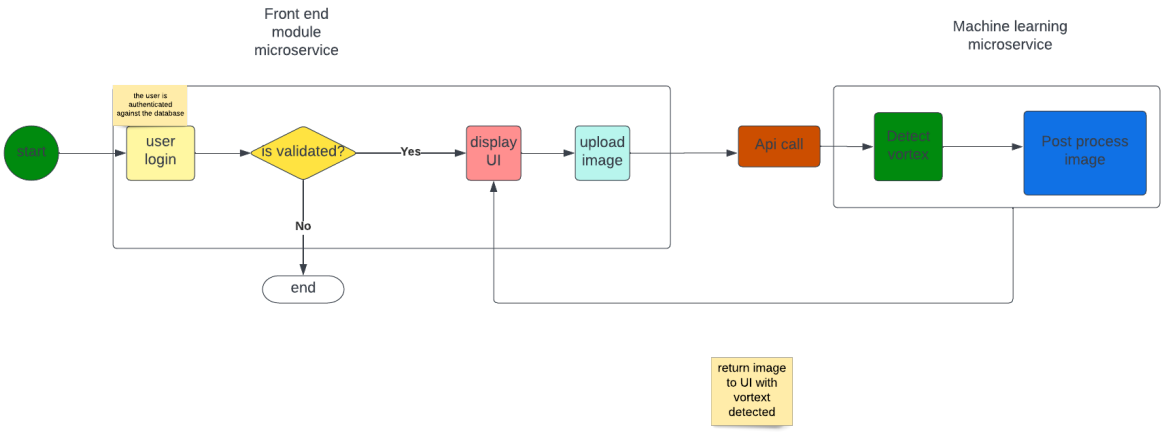



Figure 59: Flow Diagram for VvortexX Software

## B.2 Overall Description

### B.2.1 Product Perspective

This is a new product owned by Vvortexx company. It is not part not a component of a larger system; it is a standalone product, new and the first of its version. It is designed using a micro-services architecture, separating concerns of each functionality into modules. Micro-services is an architectural building a distributed application using containers (What is Micro-services? Micro-services Definition and Related FAQs — Avi Networks). A vortex detection system consists of the following components:

- An input component: This typically is the front-end user interface that takes images and image sizes as input parameters.

- A processing component: This consists of the CMV module. The CMV module pre-processes the images highlighting the area where the vortices are located.

- A machine learning module: This will have a vortex training model and a vortex detection model.

- A repository component: This component stores the processed data. For this project, a database component will be implemented and integrated into the application.This will store the information for each simulation and user profiles.

- An output component: This component typically displays the processed data. In this project,the output component is the user interface. It is a web user interface.Insert entity relationship and class diagram here.

### B.2.2 Product Functions

The system will perform several functions outlined below.

- Upload image/video This function allows the user to upload an image or video with vortices. The image will be validated against size. If the size exceeds a certain limit, it will not be uploaded.

- A REST Api call to the ML module At the backend. This call is made to detect the vortices This function detects the location of the vortices.

- Thereafter, a call is made to post process the image. This post processing phase locates the vortex cores. There are several trained models that will be used for vortex detection :
  - YOLO model
  - Convoluted neural network

- Return image with vortex location The image is returned to the UI with the vortex location.
- Create user profile. Users are created within the database. There are two kinds of users :
  - Administrators: These users are responsible for managing the overall functionality of the system.
  - Operations: These users are the CFD engineers that design aerodynamic objects.

## B.3 User Classes and Characteristics

The users of the system should be able to create other users, upload images from any source, on premises or cloud and perform general systems administration of the software. The system will have two types of user privileges, administrators, and designers. Administrators will have access to carry out all the functions within the software including vortex tracking, vortex detection and general systems administration. While the designers will be able to only upload images, track vortices and detect vortices.

Users for this application are:

- Operations users which will be mostly CFD engineers within a company that manufactures aerodynamic objects.
  - The user creates a model of the golf ball(this is applicable to any aerodynamic object) and simulates vortex generation by spinning the ball at different velocities, producing several images with vortices, generated during the simulation.
  - The web application will be launched by the user over a browser. He will have the option to upload image or video, track vortex and vortex core.
  - The user is presented with an option to track or detect the vortex.
  - The result of the vortex and vortex core is displayed on the user interface.
- Systems Administrator within a company that manages the system
  - This user is responsible for the management of the whole system.
  - He can create new system users and delete users.

10.

| User | Functionality |
|---|---|
| SYS | Creates new users,generates report,perform simulations |
| Operations | perform simulations, track vortices, upload images |

Table 10: Table 1.1 User Characteristics Table

## B.4 Operating Environment

The vortex detection system is a web application and as such, will be operated across multiple browsers like, Google chrome, Internet explorer and Microsoft edge. Operating environment for the vortex detection software is listed below:

- Software Components
  - Database: MongoDB
  - Platform: Python, react JS, Flask
  - Server: Heroku cloud
- AI Components

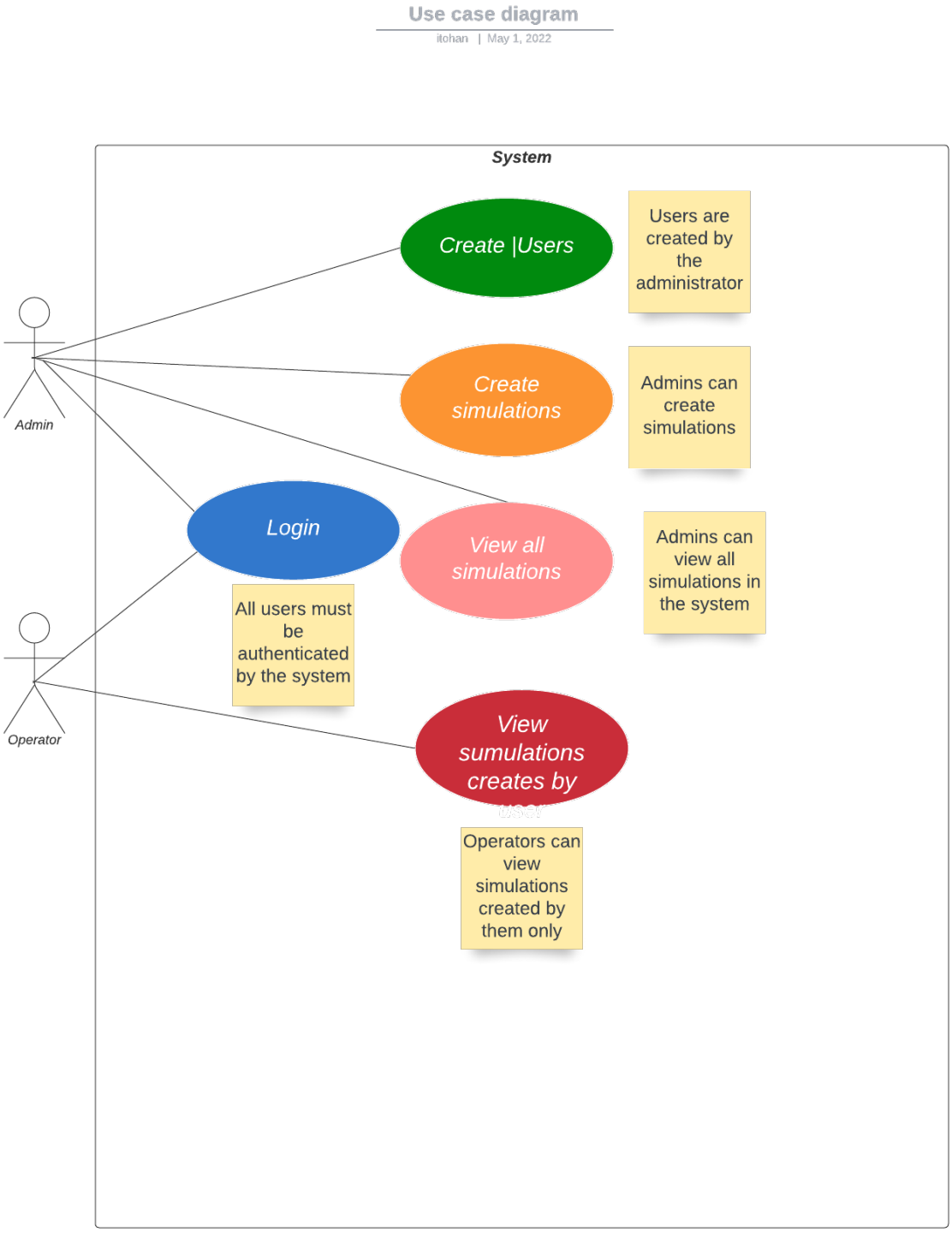



Figure 60: Use-Case Diagram

  - Platform: Python, Tensor flow
  - Server: Heroku cloud

- CFD Components
  - Fluent
  - Cartier

### B.4.1 Design and Implementation Constraints

The following are the constraints of this software:

- Software Components The software must be developed using an object-oriented programming language like C++ or Python
- There must be an API which links the front-end user interface with the back-end machine learning module
- The architecture must be a - architecture and each component of the system must be deployed separately in different containers or virtual machines
- The system must consist of a front-end user interface module and a back-end machine learning module

### B.4.2 Assumptions and Dependencies

We are assuming that this is a micro-services based architecture and all services will be deployed independently of one another. The front end will be decoupled from the back-end machine learning module. It is assumed that the data to be processed will be generated outside the vortex detection system. The images will be generated using Ansys and uploaded to an external repository on cloud, e.g., drop box, external hard drive or local hard drive. The vortex detection system needs the following dependencies:

- A database system to store simulation information
- Flask and python to develop the software
- Tensor flow framework
- YOLO is a series of deep learning model for fast object detection
- A cloud instance to deploy and run the application on cloud

## B.5 External Interface Requirement

The user interface for this system is a web user interface which is accessible over via a browser over a network. It is developed using python, react JS and flask.

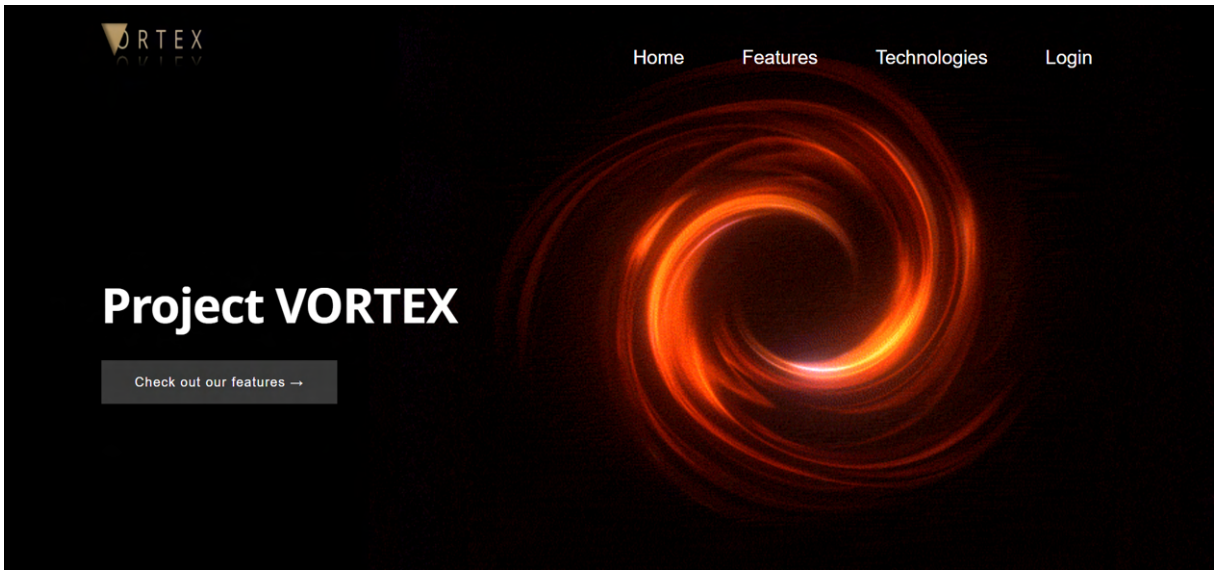



Figure 61: Front-end User Interface

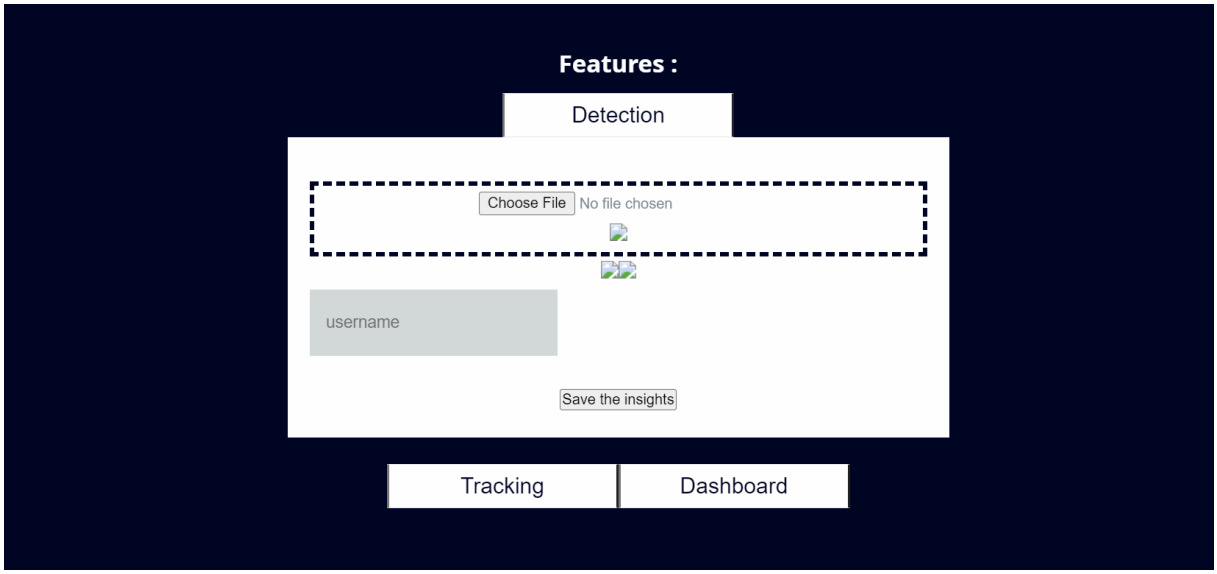



Figure 62: Front-end User Interface

### B.5.1 Hardware Interface

- Windows /Linux instances
- Browser which supports HTML and JavaScript

### B.5.2 Software Interface

Below are the software's used:

- Linux  We have chosen linux because It is robust
- Python  We have chosen python programming language because of its simplicity
- Github  We have chosen github as our code repository because it serves as a code repository that can be easily integrated into third party tools
- DE Oxygen  We chose de-oxygen because it effectively documents code and functions

# C System Features

## C.1 Upload Image

### C.1.1 Description and Priority

The vortex detection software locates and track vortices within an image, stores the data of the simulation within a database, keeps track and records of users that carry out simulations and compares the number of vortices produced by two different images. This function is of a high priority of 10 because there needs to be a medium through which the images will be uploaded into the system.

### C.1.2 Stimulus and Response Sequence

- Launch application in browser
- Login
- Select upload images
- Locate image
- Click upload button

## C.2 Track Vortex

### C.2.1 Description and Priority

Tracking the vortices with ordinary human eyes for a lot of images can be very time consuming and ineffective. This feature is required to effectively locate the vortex and has a priority of 10.

### C.2.2 Stimulus and Response Sequence

- Launch application in browser
- Login
- Select upload images
- Locate image
- Click upload button
- Click track vortex

## C.3 Upload Image

### C.3.1 Description and Priority

The vortex detection software locates and track vortices within an image, stores the data of the simulation within a database, keeps track and records of users that carry out simulations and compares the number of vortices produced by two different images. This function is of a high priority of 10 because there needs to be a medium through which the images will be uploaded into the system.

### C.3.2 Stimulus and Response Sequence

- Launch application in browser
- Login
- Select upload images
- Locate image
- Click upload button

## C.4 Create Users

### C.4.1 Description and Priority

For proper audit log, users will be created to track their activities within the system

### C.4.2 Stimulus and Response Sequence

- Launch application in browser
- Click create user
- Enter username
- Enter password
- Click save

## C.5 Functional Requirements

These are the components of the system required to make it work .

### C.5.1 Machine Learning

In order for the software to be fully functional, a machine learning module is required. This is needed to detect the vortices and vortex core.

### C.5.2 Front-end User Interface

The software must consist of an interactive user interface through which images can be uploaded .

### C.5.3 Computer Machine Vision

This functionality is required to locate the vortex cores.

### C.5.4 Database

A database system is required to persist data for simulations and retrieve reports for simulations.

## C.6 Non-Functional Requirements

### C.6.1 Performance Requirement

The minimum performance requirements for this software are

- 2GB RAM
- 4GHz CPU
- 10GB Storage

### C.6.2 Security Requirement

The security requirement for this product is that the passwords must be encrypted at login.

### C.6.3 Software Quality Attributes

- Availability : The software application should be available whenever it is needed by the user.
- Correctness : The software should accurately locate and track vortices and vortex cores.
- Maintainability : The code should be properly documented and commented. Any changes made to the code baseline should be properly documented. The code will be maintained in repository.
- Usability : The software should be able to process both image and video data.

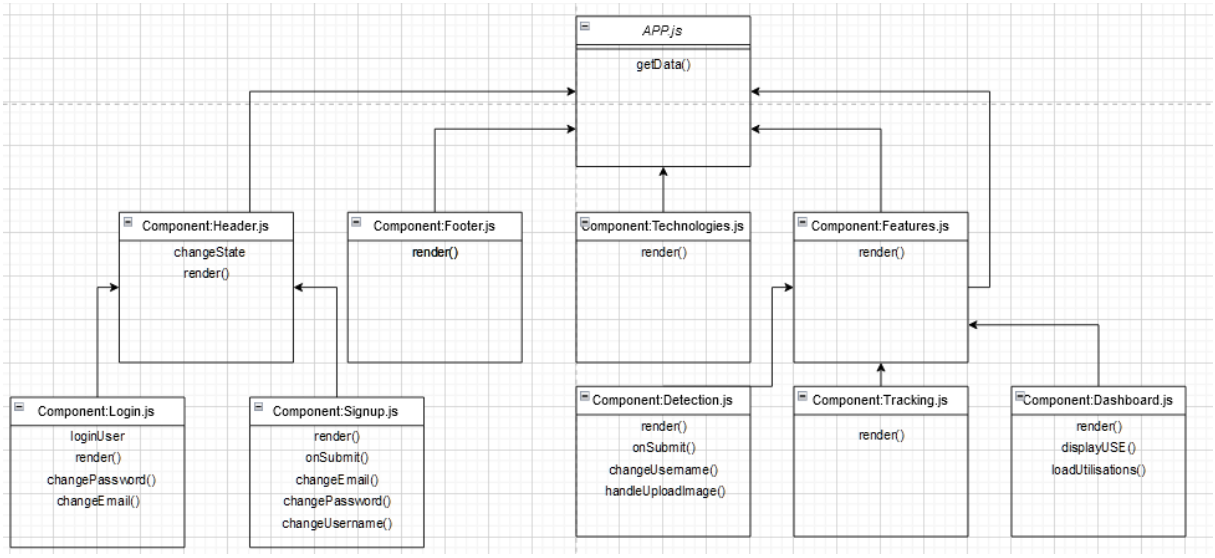



Figure 63: Software Class Diagram

# D Software Design and Architecture

## D.1 Purpose

This design document provides a comprehensive overview into the structure of the vortex detection software.It depicts the architectural view into different components of the system.

## D.2 Scope

This document gives an architectural overview of the Vortex detection software. The software is being developed to detect vortices generated by an aerodynamic object .

The document has been directly generated using Vortex detection Software Requirements and Analysis document Most part of the sections have been generated from the software requirements analysis document in section C.

## D.3 Architectural Representation

The architecture is represented as a series of UML diagrams comprising of flow diagrams, use diagrams, class diagrams, entity relationship diagrams and process view.The UML diagrams were made using Lucid Chart .

## D.4 Architectural Goals and Constraints

Key requirements and constraints that have an impact on the architecture of the system:

- All the functionality of the system must be accessible over the internet
- The vortex detection system will be implemented as a micro-services architecture. The different modules will be deployed as micro services on Heroku cloud.
- All the performance requirements as stated in the software requirements document in section B must be taken into consideration.

## D.5 Use-Case View

Below is a description of the use-case view of the software architecture. This describes the functionalities of the different components. The use-cases are initiated by the users; administrator and operator. The vortex detection system use-cases are :

- Login
- Track vortex
- Upload image

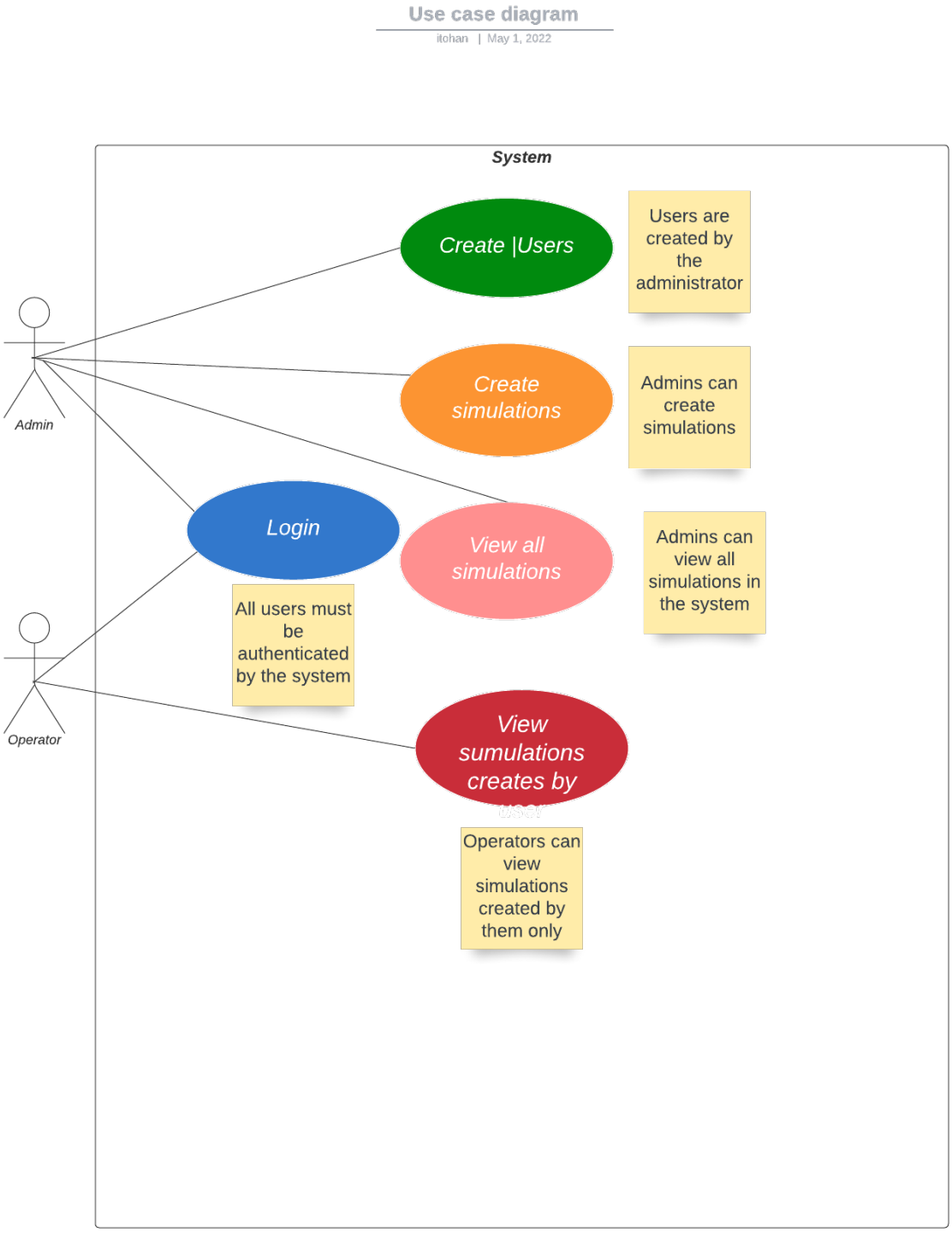



Figure 64: Significant Use-Case

- Create users
- View simulation dashboards

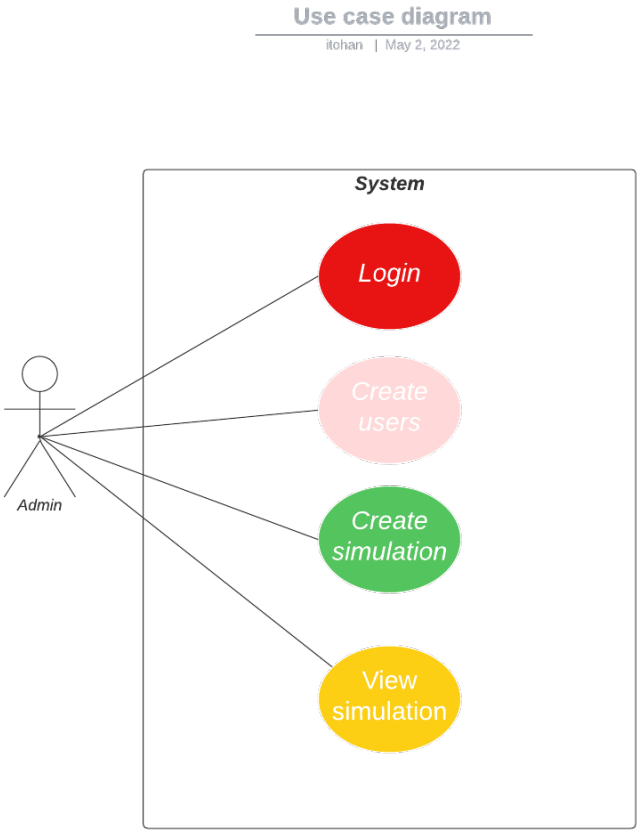



Figure 65: Use-Case for Admin

### D.5.1 Architecturally-Significant Use-Case

This use-case diagram shows the activities of each user within the system.

- Login Brief Description: This is the user authentication module of the system.
- Track vortex Brief Description: This is the machine learning module of the system.

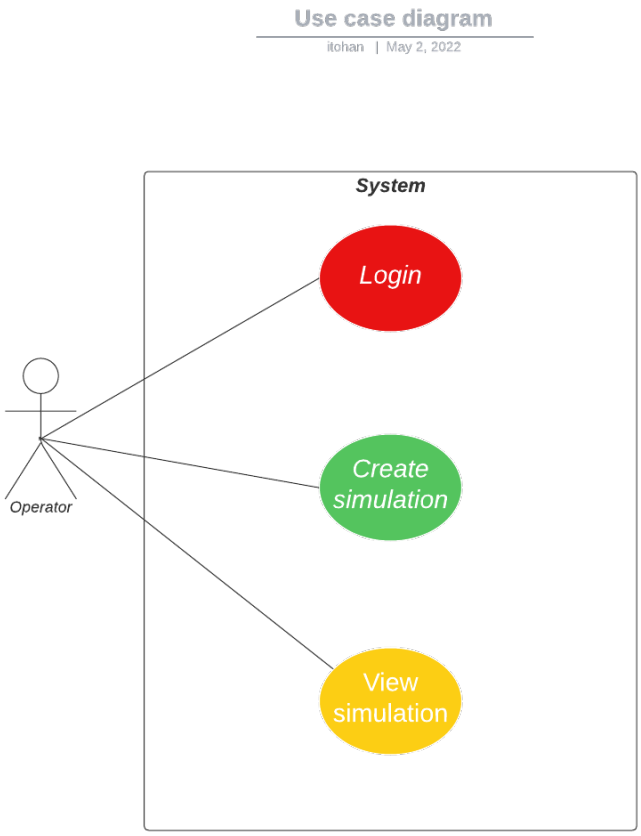



Figure 66: Use-Case for Operator

It uses machine learning to accurately locate the vortices.

- Upload image Brief Description: This is part of the front-end part of the system. Data to be processed is uploaded via the front end .
- Create users Brief Description: This module creates user profiles into the system.
- View simulation dashboards Brief Description: This allows the users to view com-

pleted simulations.

## D.6 Logical Overview

This is a logical overview of the system architecture of vortex detection system.The most important classes and subsystems is described. These relationships are depicted using class diagrams.

The logical overview of the system comprises of three main aspects :

- The application layer which comprises of the front end
- The business logic layer, which comprises of business rules.
- The application programming interface layer which connects the machine learning module with the business logic
- The database objects

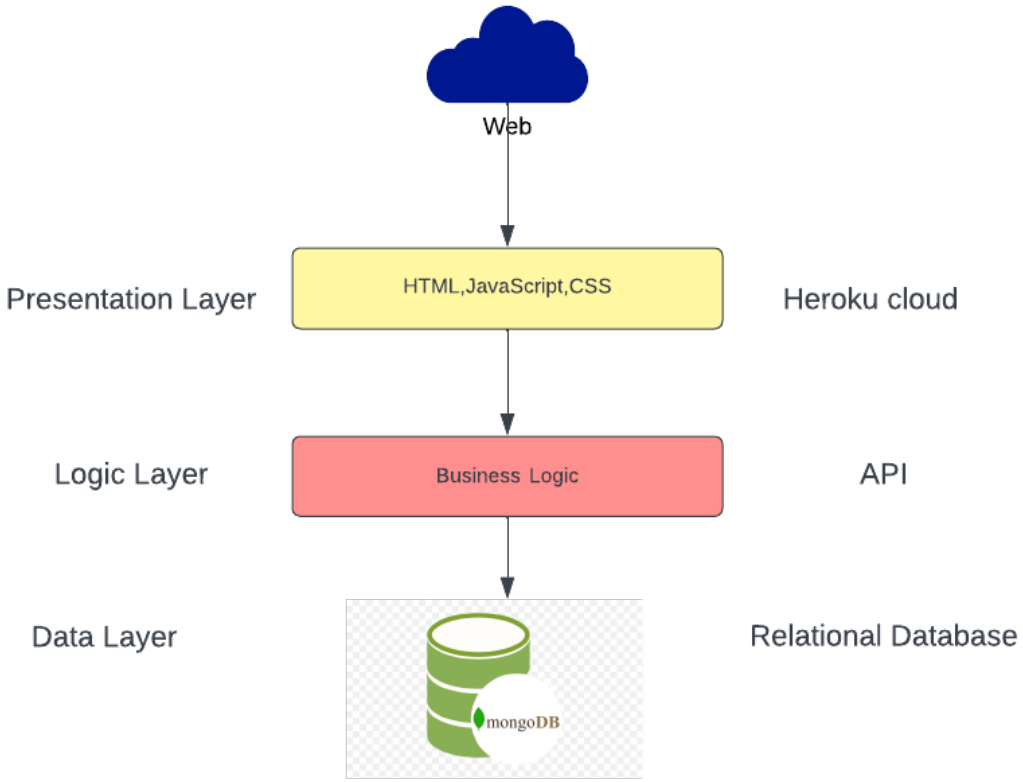



Figure 67: Logical Diagram

### D.6.1 Application Layer

The application layer has all the classes that represent what the user sees and interacts with .

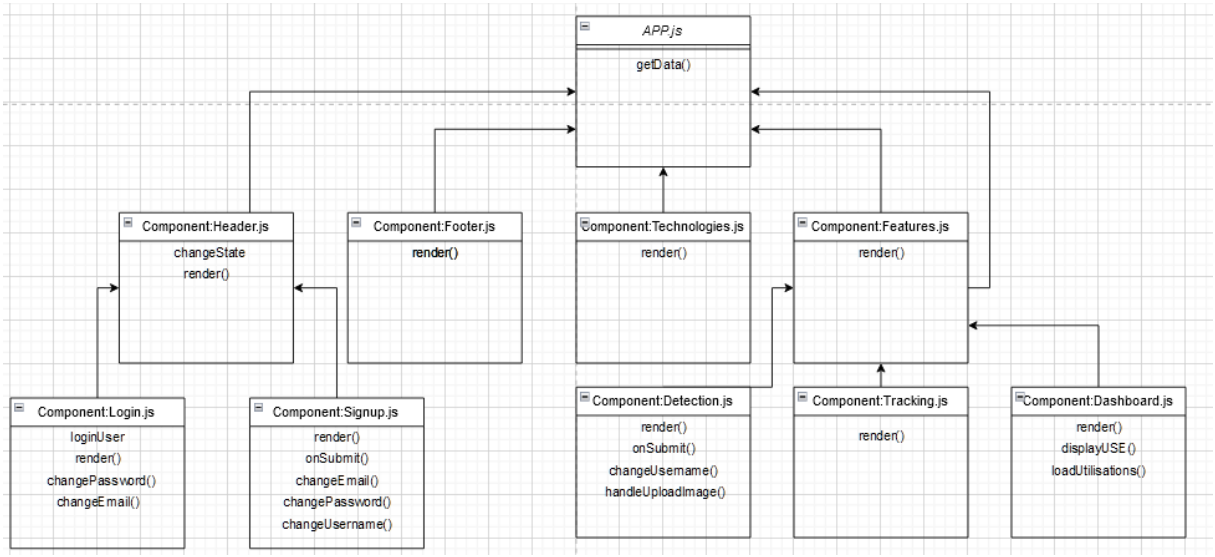



Figure 68: Front-End Class Diagram

### D.6.2 Business Logic Layer

This layer contains the application programming interface logic.

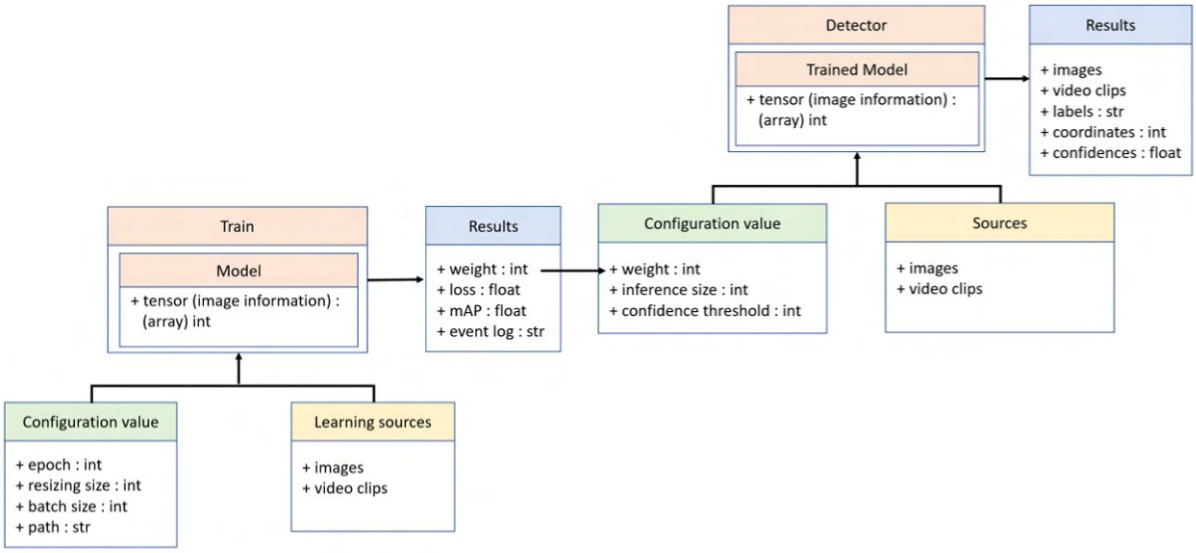



Figure 69: Business Layer Class Diagram for Vortex Location

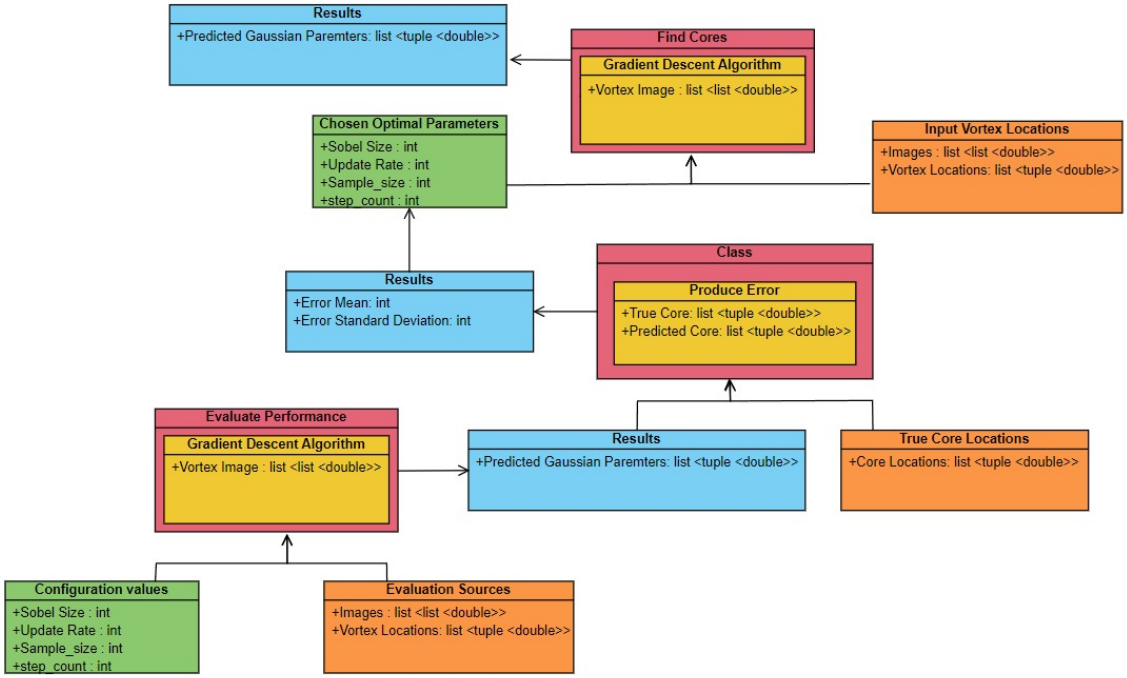



Figure 70: Business Layer Class Diagram for Vortex core Location

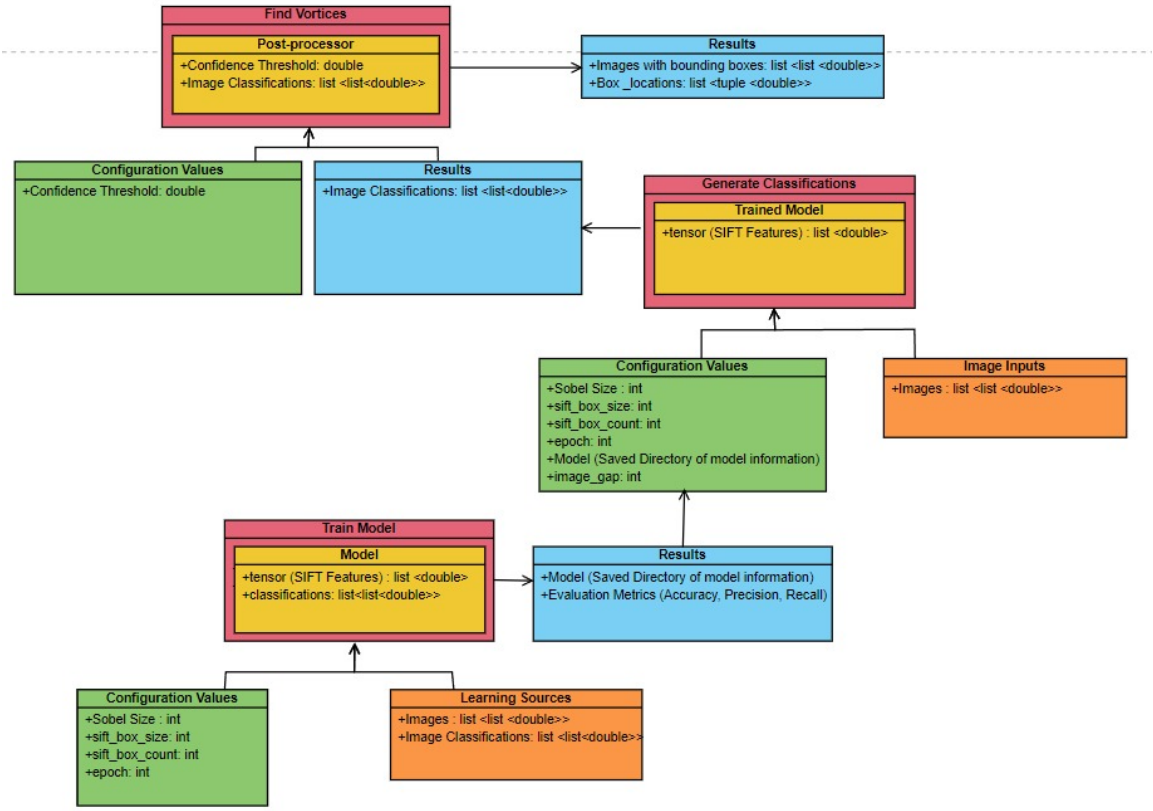



Figure 71: Business Layer Class Diagram for Vortex core Location SIFT Method

## D.7 Data Layer

This layer consists of the database entity.

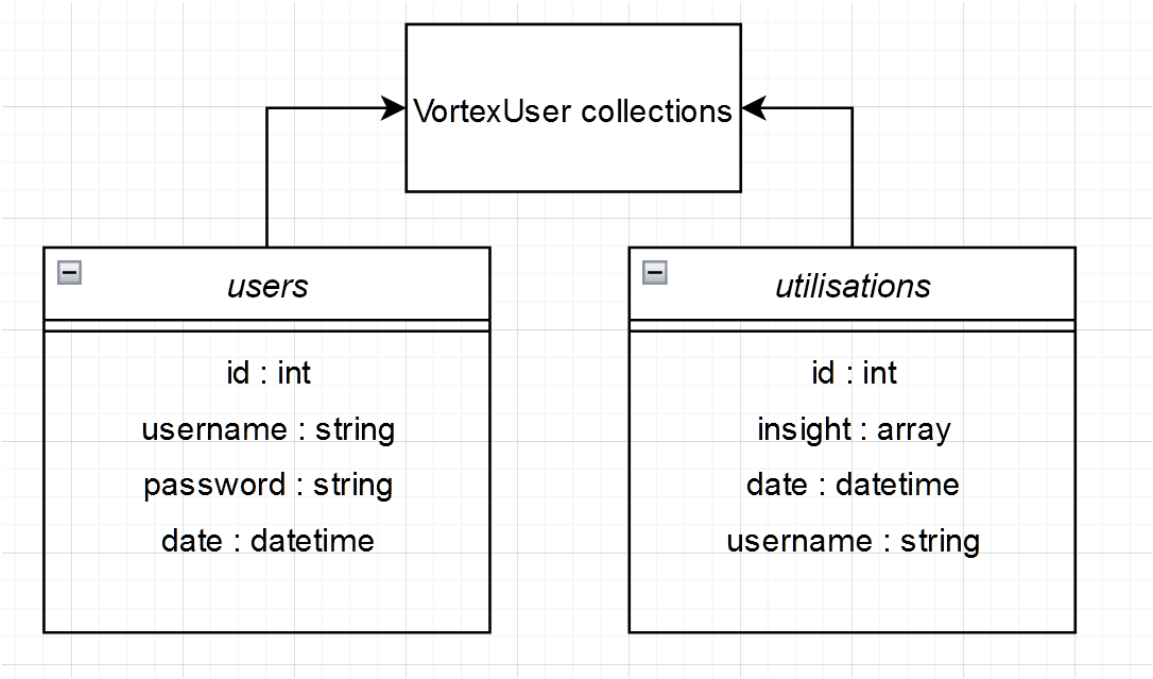



Figure 72: Data Layer Entity Relationship Diagram

## D.8 Deployment Architecture

The deployment architecture is micro-services. A micro-service architecture loosely couples the components of a software. All the components are connected via an API.

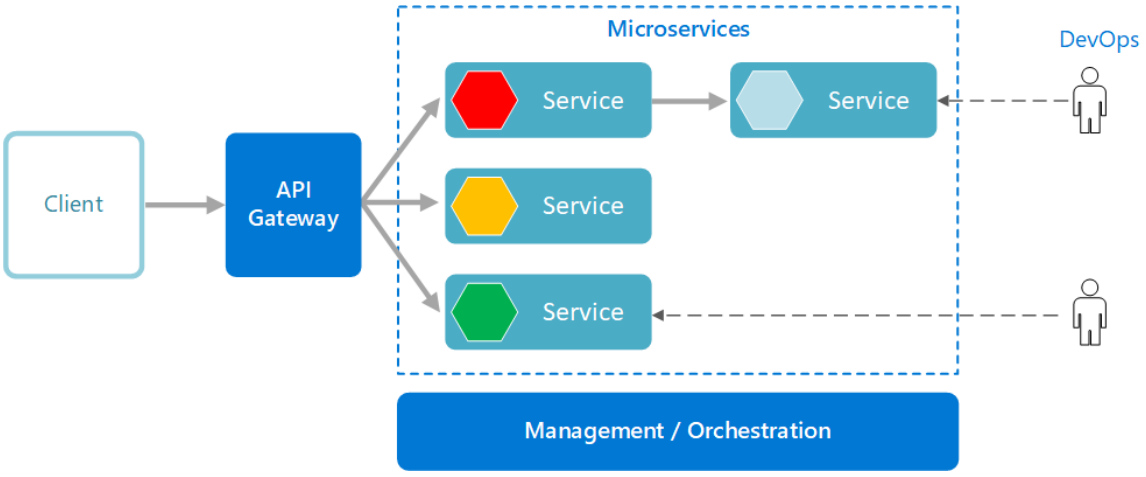



Figure 73: Microservices Architecture

## D.9 Process View

The process view shows how all the components of the application are connected.The data is created and processed externally by the CFD engineer :


Figure 74: Mesh Design Process

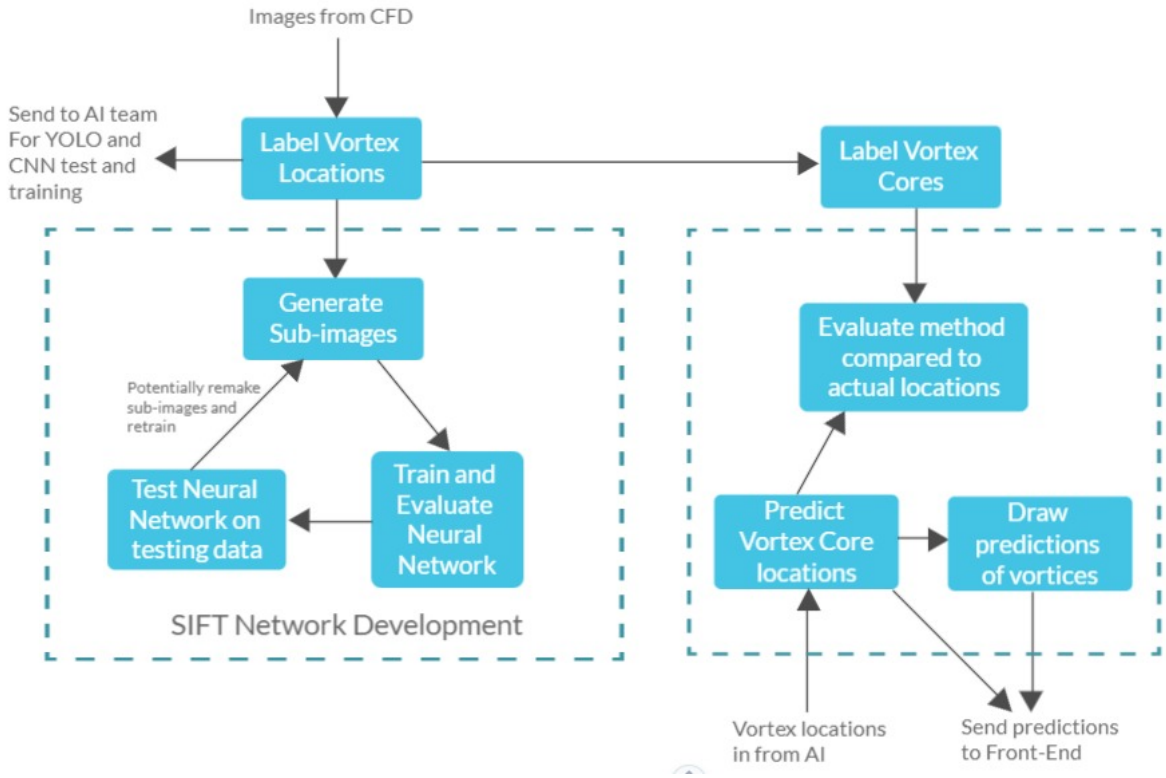



Figure 75: System Internals

# E Risk Analysis Document

CRANFIELD UNIVERSITY

Software Engineering for Technical Computing MSc (*Risk Study*)

**Group Project**

VvortexX

**Investigation of Vortex Behaviour for Practical Application**




Pierre Emmanuel-Désiré Cissé (S368793)

# 1-Risk Assessment Matrix

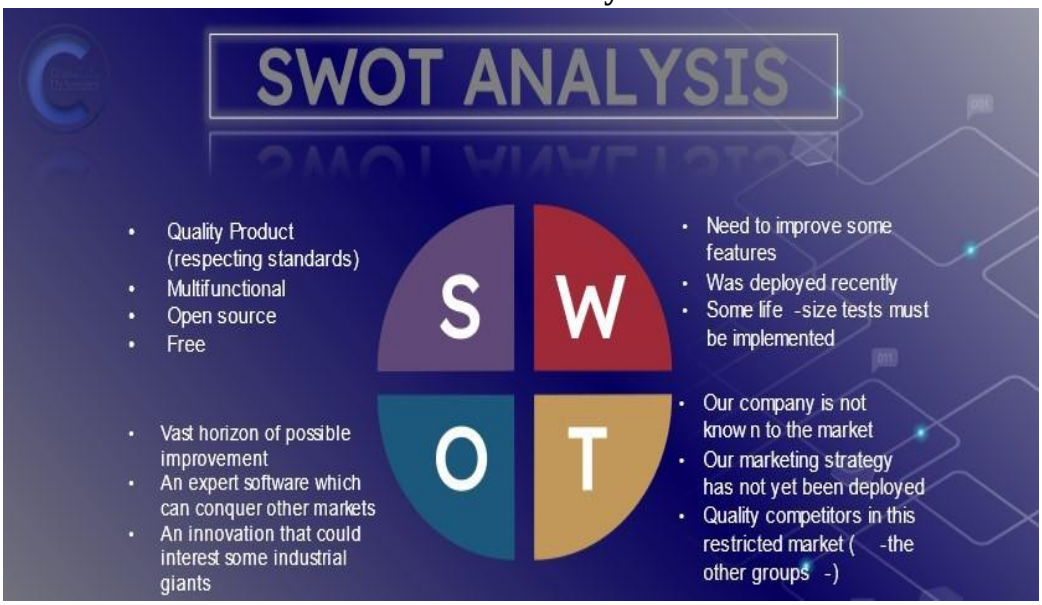
Risks in project management are unanticipated events that may or may not occur and have an impact on the outcome of your project. Analyzing and managing risks is a critical practice in project management, according to the Project Management Institute (PMI). It increases the likelihood of a successful project completion while minimising the consequences of any risks that may arise.

| NAME | Pierre Emmanuel-Désiré CISSE | OBJECTIVE | EXISTENTIAL RISK ASSESSMENT |
|---|---|---|---|

| REF / ID | PRE-MITIGATION | | | | DEPARTMENT / LOCATION | MITIGATIONS / WARNINGS / REMEDIES | POST-MITIGATION | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | RISK | SEVERITY OF RISK | RISK PROBABILITY | RISK LEVEL | | | SEVERITY OF RISK | RISK PROBABILITY | RISK LEVEL | AGREE TO GO FORWARD? |
| | | – Insignificant<br>– Marginal<br>– Moderate<br>– Critical<br>– Catastrophic | – Definite<br>– Likely<br>– Occasional<br>– Seldom<br>– Unlikely | **- LOW**<br>**- MEDIUM**<br>**- SERIOUS**<br>**- HIGH**<br>**- EXTREME** | **- CFD**<br>**- AI**<br>**- SE** | | – Insignificant<br>– Marginal<br>– Moderate<br>– Critical<br>– Catastrophic | – Definite<br>– Likely<br>– Occasional<br>– Seldom<br>– Unlikely | **- LOW**<br>**- MEDIUM**<br>**- SERIOUS**<br>**- HIGH**<br>**- EXTREME** | **YES / NO** |
| R01 | Management – Latency, Significant time to get used to the team | Marginals | Definite | **LOW** | CFD & AI | Tutorials | Insignificant | Unlikely | **LOW** | **YES** |
| R02 | New tools – no effectiveness | Moderate | Definite | **LOW** | ALL | Workshops | Insignificant | Unlikely | **LOW** | **YES** |
| R03 | Additional Requirements, system requirements are not achieved | Critical | Occasional | **HIGH** | AI & SE | Prioritize the goals | Marginal | Seldom | **MEDIUM** | **YES** |
| R04 | Addition of unknown features | Insignificant | Occasional | **LOW** | SE | Concession about certain choices | Insignificant | Unlikely | **LOW** | **YES** |
| R05 | Inadequate design, minor decrease in performance (Redesign required) | Marginal | Occasional | **HIGH** | AI | Redesign | Insignificant | Seldom | **MEDIUM** | **YES** |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| R06 | Overly optimistic schedule, the scope of the project might change | Critical | Likely | **SERIOUS** | ALL | Sprint 2 : reschedule | Marginal | Unlikely | **MEDIUM** | **YES** |
| R07 | Dissensions in the team | Moderate | Occasional | **EXTREME** | ALL | Discussion about issues | Insignificant | Seldom | **SERIOUS** | **YES** |
| R08 | Lack of cooperation and commitment of some team members | Critical | Seldom | **EXTREME** | ALL | Test different communication strategies | Insignificant | Seldom | **SERIOUS** | **YES** |
| R09 | Testing reveals problems | Marginal | Occasional | **SERIOUS** | AI | Retrain the model with new samples | Insignificant | Unlikely | **MEDIUM** | **YES** |
| R10 | The new release of the software is not working | Catastrophic | Seldom | **EXTREME** | SE | Use git to go back to the previous version | Insignificant | Unlikely | **SERIOUS** | **YES** |
| R11 | Lack of external users to test the prototypes | Marginal | Definite | **MEDIUM** | SE | Each member test | Marginal | Unlikely | **LOW** | **YES** |
| R12 | Lateness regarding the milestones | Critical | Seldom | **SERIOUS** | ALL | Speed-up some outputs | Marginal | Unlikely | **MEDIUM** | **YES** |
| R13 | A slowdown in enthusiasm and overall motivation | Moderate | Occasional | **SERIOUS** | ALL | More workshops and standup meetings | Insignificant | Unlikely | **MEDIUM** | **YES** |

## 2-Market Analysis



# F Quality Strategy Document

## F.1 Test Plan Documents

CRANFIELD UNIVERSITY

Software Engineering for Technical Computing MSc (*Quality Document*)

**Group Project**

VvortexX

**Investigation of Vortex Behaviour for Practical Application**




Pierre Emmanuel-Désiré Cissé (S368793)

# VvortexX COMPANY

VvortexX 1.2

# QUALITY STRATEGY

Version 1.2

April 2022

*Pierre Emmanuel-Désire CISSE (SE&CFD Engineer/ Leader of VvortexX, Author of this document)*

## *1. Test Plan Identifier*

Vortex-Website 1.2 release 1.1


Note: the structure of this document is primarily based on the IEEE 829-1998 Standard for Software Test Documentation. Additional reference standards include IEEE 1008 (Unit Testing), 1012 & 1059 (Validation & Verification), and 1074 (Software Life Cycle process).

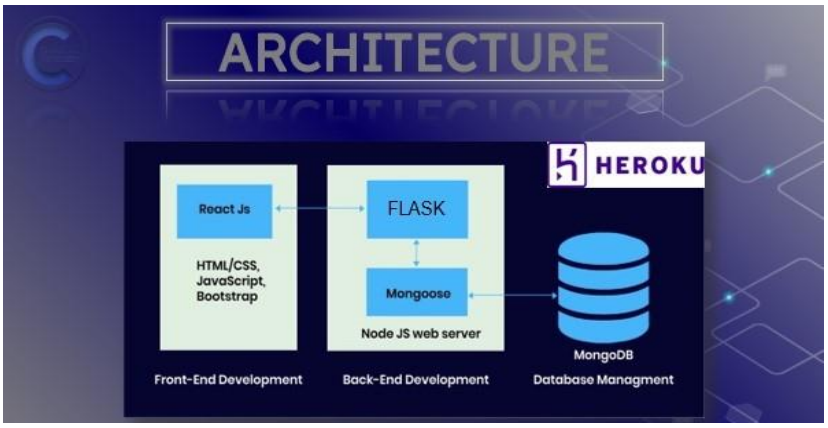
## *2. Introduction*

VvortexX Company is a start-up born and driven by the passion of a few experts. The launch project, current main project of the company, is the creation of a website for the Investigation of Vortex Behavior for Practical Application.

The website aims in its first version to analyze on images/videos the presence of vortices and their aerodynamic behavior. It is specially designed for optimal precision in the study of the fluidic behavior of moving objects (i.e golf balls) in order to facilitate the industrial design work of this type of object. Processed files from CFD simulations: the website is an expert market site (i.e. computational design department of a design company) that wants to automate some calculations in order to make judicious design choices (i.e shapes choices). The initial release of the software will be known as Vortex-Website 1.2 release 1.1.

The target audience will initially be Computational Design experts (or students) with an interest in the study of the fluidic behavior of their simulations with regard to the vortices' presence. Once the system is working successfully, VvortexX Company intends to develop the site with regard to the features (which are in the initial version: detection and tracking) and the functionalities of the site (which are in

the initial version: user account management and simulation report). The system is developed in with this structure, for the sole purpose of achieving the company's quality objectives:




Due to pressing business needs, CU's traditional development and testing processes have been customized to allow for faster and more frequent delivery cycles. Specifically, testing will now consist of the following phases (not listed chronologically):

- Unit and integration level – adherence to coding standards and successful communication between units
- Code Quality Assurance - acceptance into system-level testing by successfully repeating a small subset of the tests performed in the code and integration level
- System-level – performance, functionality etc.
- System Quality Assurance & Acceptance (acceptance into Production)
- Post Implementation

Each testing phase will be described within its own test plan (included as Appendixes to the Master Test Plan). While the MTP (this document) may outline the overall strategy and document the aspects of the testing that are common to all the phases.

Keeping in mind the multipolar aspect of the website: Computational Fluid Dynamics, Artificial Intelligence, Software Engineering. The site will host (back-end) an ML model from data collected via Fluent & Paraview simulations, thus three (3) teams are working together on the project (CFD, AI, SE). As a result, the quality strategy is decoupled into three and this document summarizes the overall vision of it (each test step is necessary for the three departments). The canons of the quality strategy are defined in large team and the author of this document is the Quality/ Test Manager: but the test logs (once defined) are executed by the programmers and monitored by the manager who makes sure of their validity in an Agile way. The total team consists of:

- Daniel Smythe (DS) – AI – Computer Machine Vision Engineer
- Vetrichelvan Pugazendi (VP) – CFD – Computational Design Engineer
- David Wang (DW) – SE – Software Engineer/ Programmer
- Itohanoghosa Eregie (IE) – SE – Software Engineer
- Tae Yong Kim (TK) – AI – Machine Learning Engineer
- Imad Ui Islam (II) – AI – Machine Learning Engineer
- Pierre Cisse (PC) – SE&CFD – Software Engineer/ Project Leader – Quality Manager

## *3. Test Items*

The scope of this Testing activity will include:

- The system software and supporting infrastructure
- The design of the model (or models) used in the back-end of the system software

The scope of this testing activity will not include:

- Vortex-Website 1.2 release 1.1 's documentation e.g.: Requirements & Design Specifications or User, Operations & Installation Guides
- Any other CU applications
- Any Legacy systems that the Vortex-Website 1.2 release 1.1 software integrates with (with the exception of the interface)

## *4. Features and Functions to Test*

Testing will consist of several phases (see introduction), each phase may or may not include testing of anyone or more of the following aspects of the Vortex-Website 1.2 release 1.1:

- Coding standards: compliance with norms (programming)
- Content: compliance with validation samples (CFD, AI)
- Functionality: expected functioning of features
- Verifiability (Understandability): properly commented and documented (if applicable)
- Efficiency (Performance): a certain level of accuracy is decided for CFD and AI models
- Reliability (Behave properly, As expected): reaction to some end-user behavior (wrong entries...)
- Portability (To new environments/ platforms): dockerize the subsystems, use API for communication between systems
- Evolvability (Easy to change, Adaptability, Repairability): use intuitive frameworks, languages, packages for the targeted fields since the project is currently opensource
- Robustness (In none-specified circumstances): automate software deployment and test application
- Usability (Error conditions, User-friendliness): UX-improvement

## *5. Features and Functions Not to Test*

It is the intent that all of the individual test cases contained in each test plan will be performed. However, if time does not permit, some of the low-priority test cases may be dropped.

## *6. Approach/Strategy*

The philosophy of the testing is risk-based testing, i.e. each test case will be prioritized as, High, Medium, or Low priority and then scheduled accordingly (Highest first). Exceptions to this general rule might include instances where:

- A large number of low-priority test cases can be executed using a small number of resources • Scheduling conflicts arise.
- A lower priority test is a pre-requisite for another higher priority test.
- A risk alert from a risk assessment meeting that generates a new priority issue to be fixed.

The testing will use a combination of manual and automated testing, due to the limited duration of the testing; only automated tools that are already familiar to the CU staff or have a minimum learning curve will be used.

Basic metrics will be kept for test effort (i.e. hours), test cases executed, and incidents. Due to the lack of available tools and time, no attempt will be made to collect more sophisticated metrics here such as code coverage.

## *7. Item Pass/Fail Criteria*

The entrance criteria for each phase of testing must be met before the next phase can commence. Formal approval will be granted by the responsible for each team (CFD, AI, SE) in agreement with the Quality/ Test Manager.

## *8. Suspension Criteria and Resumption Requirements*

In general, testing will only stop if the Vortex Website becomes unavailable. If testing is suspended, it will be resumed once access to the software is reestablished.

Certain individual test cases may be suspended, skipped or reduced if prerequisite tests have previously failed.

## *9. Test Deliverables*

The following documents will be generated as a result of these testing activities:

- Master test plan (MTP - this document)
- Individual test plans for each phase of the testing cycle (as an Appendix to the MTP)
- Combination incident/test summary reports for each phase
- Test log for each phase
- Automated test scripts and supporting test data

Note: Under normal testing conditions, a daily incident report would be produced rather than a combined incident/test summary report. A daily incident report would normally be used by the development team to get a "heads up" on any potential defects and allow a Change Control Board (CCB) to decide which enhancements/fixes could be introduced into the test environment during the current testing cycle. However, due to the short test execution time period a single incident/test summary report will be produced for each of the testing phases.

## *10. Remaining Test Tasks*

Upon delivery of the test deliverables and the successful implementation of the Vortex-Website 1.2 release 1.1, all of the tasks covered by this master test plan will be deemed to have been completed. The only exception being the post-implementation test plan, which will be a continuing effort until the application is replaced or decommissioned for the sake of real robustness.

## *11. Test Environments*

As detailed above, the tools used will depend on the type of test carried out (CFD, AI, SE): the choice of tools is therefore left to the discretion of those concerned who must nevertheless refer to the Quality/Test Manager to validate the choices and possible change.

## *12. Responsibilities*

The following people will be responsible for:

- Daniel Smythe– Implement the tests defined by the Manager (test log) in CMV-model
- Vetrichelvan Pugazendi– Implement the tests defined by the Manager (test log) in CFD-model
- David Wang– Implement the tests defined by the Manager (test log) in the prototype (final software)
- Itohanoghosa Eregie– Ensure the proper use of test tools and centralize tests (via CircleCI)
- Tae Yong Kim– Implement the tests defined by the Manager (test log) in ML-model
- Imad Ui Islam– Implement the tests defined by the Manager (test log) in ML-model
- Pierre Cisse– Define and Lead/ Manage the Quality Strategy

## *13. Staffing and Training Needs*

The relevant CU managers will ensure that the staff assigned to this project are experienced with:

- General development & testing techniques
- CU's development lifecycle methodology
- All development and automated testing tools that they may be required to use

### *14. Schedule*

The following tentative schedule will hopefully be met:

- Test design (this document) is expected to be fully completed by the end of the first month (final version)
- Test execution is expected to be implemented in an agile way throughout the project
- Producing the Test Incident/Summary report is expected to be completed within 1 day of completing 1 test execution phase

A more detailed breakdown is currently being developed in JIRA project management tool and will be completed before this master test plan is approved.

### *15. Risks and Contingencies*

The following seeks to identify some of the more likely project risks and propose possible contingencies:

- Software becomes unavailable – Testing will be delayed until this situation is rectified - May need to call on other members of the company but from other teams to do the testing or reduce the number of test cases.
- Testing tool is not available/does not work - This will delay the introduction of automated testing and result in more manual testing - May need to call on other members of the company but from other teams to do the testing or reduce the number of test cases.
- A large number of defects/incidents makes it functionally impossible to run all of the test cases – As many test cases as possible will be executed.
- Not enough time to complete all test cases. If time cannot be extended, individual test cases will be skipped, starting with the lowest priority.

### *16. Approvals*

The Q/ T Manager and the staff must approve this plan.

## Appendix

This Appendix is comprised of the following test plans (listed chronologically):

**Unit and Integration Level**
**Code Quality Assurance (acceptance into System testing)**
**System Level**
**System Quality Assurance & Acceptance (acceptance into Production)**
**Post Implementation**

# Release 1.1
# UNIT & INTEGRATION TEST PLAN

Version 1.2

April 2022

*Pierre Emmanuel-Désire CISSE (SE&CFD Engineer/ Leader of VvortexX, Author of this document)*

## *1. Test Plan Identifier*

Vortex-Website 1.2 release 1.1.UITP 0.1

Note, the structure of this document is based on the IEEE 829-1998 Standard for Software Test Documentation

## *2. Introduction*

This testing phase will use a number of testing techniques: code reviews and inspections, white-box testing, “buddy” testing etc to ensure that the code matches the required program specifications. The decision as to which technique(s) to use for any given unit of code will reside with the Tester responsible for the Module.

## *3. Features and Functions to Test*

### Coding standards

Each of the units of code that make up the module being tested must be coded to all of the following coding standards, any deviations from the standard must be documented and approved

UICS1 - High

The code must pass the following syntax and design requirements:

- CU's standard naming conventions
- `{OOP Approach | PEP 8, 257 – Style Guide for Python Code}`
- Any exceptions must be documented and approved
- All development/testing source code will be documented/commented with standard module (thereby making maintenance/debugging easier), especially for server-side includes

  e.g.: using Doxygen.
- Module name
- Original author
- Date initially written
- Language and version
- CC Copyright
- Enhancements/changes log - who/when/why

  Note: For performance and security reasons, the production version (as seen by the user) will not contain this information (with the exception of the copyright, which should be placed in the "Copyright" meta tag)
- Error messages do not describe the internal workings of the program

### Content

UICN1 - High

Compliance with validation samples (CFD, AI)

UICN1 - Medium

No “hidden” text is to be used

### Functionality

UIFU1 - High

The Software implements all of the business requirements described in its’ associated requirements specifications. Acceptable functional should be checked for under the following adverse client-side conditions:

- Detection of vortex.
- Tracking of vortex.
- User-account management and dashboard for report per simulation.

### Verifiability

UIVE1 - Medium

Properly commented and documented (if applicable)

### Efficiency

UIEF1 - High

A certain level of accuracy is decided for CFD and AI models

### Reliability

UIRE1 - High

Reaction to some end-user behavior:

- During login and/or creation of an account.
- When using features (add an image instead of a video and vice versa).

### Portability

UIPO1 - Medium

Dockerize the subsystems, use API for communication between systems

### Evolvability

UIEV1 - Medium

- Being opensource

### Robustness

UIRO1 - Medium

Automate software deployment and test application via CircleCI & Git.

### Usability

UIUS1 - Medium

The application should make it easy to follow the reasoning adopted by the development team. The software must be UX-wise.

## *4. Features and Functions not to Test*

None

## *5. Test Deliverables*

Incidents and defects from this test plan will be included in the combination unit/integration incident/test summary report.

## *6. Test Environment*

The development environment will be used for all unit and integration tests.

VvortexX 1.2

# Release 1.1
# CODE QUALITY ASSURANCE TEST PLAN

Version 1.2

April 2022

*Pierre Emmanuel-Désire CISSE (SE&CFD Engineer/ Leader of VvortexX, Author of this document)*

## Code Quality Assurance Test Plan

### *1. Test Plan Identifier*

Vortex-Website 1.2 release 1.1.CQTP 0.1

### *2. Introduction*

The purpose of this phase of testing is twofold: first, as a means of ensuring the quality of the unit/integration testing process (i.e. are the team members responsible for performing the unit/integration testing following the guidelines specified in the unit/integration testing test plan) and secondly, as a means of ensuring that the next release of the application has been sufficiently tested at the Unit/Integration level and thereby is "fit enough" to warrant further testing at a higher level (i.e. the application meets the system testing phase entrance criteria).

The strategy employed for this phase can be summarized as follows:

- The Actions units will be retested using the unit/integration test plan, the results of which, should exactly match the results obtained during the unit/integration phase
- Once the original set of unit/integration tests have been validated (via the random re-testing), a decision will be made as to whether the discrepancies (if any) between the original results and the re-test results, indicate a failure in the unit/integration testing process. If the head of QA determines that the unit/integration process is operating in a less than optimal manner, then the head of QA will call a meeting of the project review board to discuss what actions should be taken to improve the situation

### *3. Features and Functions to Test*

The scope of this phase of testing will include any of the units that comprise the Vortex-Website 1.2 release 1.1.

### *4. Features and Functions not to Test*

No attempt will be made to review the comprehensiveness of the test cases specified in the unit/integration testing test plan or the desirability of the coding practices specified in the development standards.

### *5. Test Deliverables*

Incidents and defects from this test plan will be included in the combination code quality assurance incident/test summary report.

## *6. Test Environment*

Where possible the same test environment will be used for code quality assurance testing as was used for unit/integration testing.

# Release 1.1
# SYSTEM TEST PLAN

Version 1.2

April 2022

*Pierre Emmanuel-Désire CISSE (SE&CFD Engineer/ Leader of VvortexX, Author of this document)*

## *1. Test Plan Identifier*

Vortex-Website 1.2 release 1.1.SYTP 0.1
Note: the structure of this document is based on the IEEE 829-1998 Standard for Software Test Documentation

## *2. Introduction*

This test plan outlines the tests that will be conducted by the testers the majority or all of the individual units that make up the Vortex-Website 1.2 release 1.1 have been initially tested and found to be free of significant error.

Rather than assigning a portion of the application (final prototype) to an individual tester and expecting them to thoroughly test all the different aspects of the software (functionality, performance etc). Testers (or teams of testers) will be expected to a single aspect of the entire application e.g. one will performance test the entire application, while another will concentrate on just functionality. This approach was selected because it was felt that this strategy would allow for more tests to be run in parallel (as the project is hosted externally) and hence would have a shorter elapsed time frame and would also potentially have a shorter learning curve for the testing group, who have had relatively little experience testing applications (multi-backgrounded company).

## *3. Features and Functions to Test*

### Functionality

SYFU1 - High
Every application is re-checked to ensure that all of the business requirements described in the specifications have been implemented correctly.

Note: in theory, these tests should not detect any new problems, as this functionality should have been performed during unit testing. However, because of the high business risk associated with the functionality of this application, the business logic used to provide the functionality of the software will be manually retested.

### Reliability

SYRE1 - High
Re-checked reaction to some end-user behavior.

### Robustness

SYRO1 - Medium
Recompile.

Usability

SYUS1 - Medium

Test the GUI to varied users.

## *4. Features and Functions not to Test*

Notable features and functions that will not be tested include.

## *5. Test Deliverables*

Incidents and defects from this test plan will be included in the combination system incident/test summary report.

## *6. Test Environment*

Via **CircleCI.**

# Release 1.1
# SYSTEM QUALITY ASSURANCE & ACCEPTANCE TEST PLAN

Version 1.2

April 2022

*Pierre Emmanuel-Désire CISSE (SE&CFD Engineer/ Leader of VvortexX, Author of this document)*

## System Quality Assurance & Acceptance Test Plan

### *1. Test Plan Identifier*

Vortex-Website 1.2 release 1.1.SQTP 0.1

### *2. Introduction*

The strategy employed for this phase can be summarized as follows:

- A risk assessment will be made of the software.
- The test cases designed to detect/mitigate any of the "high" risk features or attributes of the application will be re-tested
- Once the original set of "high" risk system tests have been validated (via the random retesting), a decision will be made as to whether the discrepancies (if any) between the original results and the re-test results indicate a failure in the system testing process. If the Q/ T Manager determines that the system process is operating in a less than optimal manner, then he will call a meeting of the project review board to discuss what actions should be taken to improve the situation

### *3. Features and Functions to Test*

The scope of this phase of testing will include any of the features or attributes that comprise the website. In time permits, all of the test cases designed to detect/mitigate any of the "high" risk features or attributes of the application will be re-tested. If time does not permit a full re-test of all of the "high" risk-mitigating tests, then a random subset of these test cases will be selected and executed.

With the exception of some ad-hoc testing performed as part of the software's acceptance, all of the test cases used in this phase will be based on test cases from the original system testing test plan.

### *4. Features and Functions not to Test*

No attempt will be made to review the comprehensiveness of the test cases specified in the system testing test plan. A separate task (initiated at project start-up) developed the testing standards and was subject to a separate review and approval process.

### *5. Test Deliverables*

Incidents and defects from this test plan will be included in the combination system quality assurance incident/test summary report.

## *6. Test Environment*

Where possible, the system quality assurance tests environment should be an exact match of the implementation environment.

# Release 1.1
# POST-IMPLEMENTATION TEST PLAN

Version 1.2

April 2022

*Pierre Emmanuel-Désire CISSE (SE&CFD Engineer/ Leader of VvortexX, Author of this document)*

## *1. Test Plan Identifier*

Vortex-Website 1.2 release 1.1.PITP 0.1
Note: the structure of this document is based on the IEEE 829-1998 Standard for Software Test Documentation.

## *2. Introduction*

Testing doesn't stop when the application goes into utilization – it's not a bounded interval event. Instead, Software testing is a regular process that continues as long as the application is live. There are many maintenance-testing activities that will need to be performed on a regular basis after the application has been implemented. This test plan outlines the tests that will be performed on a regular basis until either the product is replaced or decommissioned.

## *3. Features and Functions to Test*

Functionality
PIFU1 - High

Reliability
PIRE1 - High

Robustness
PIRO1 - Medium

Usability
PIUS1 - Medium

## *4. Features and Functions not to Test*

Non-notable.

## *5. Test Deliverables*

Incidents and defects from this test plan will be included in the combination post-implementation incident/test summary report.

## *6. Test Environment*

The production environment will be used for all post-implementation tests.

VvortexX 1.2

**Release 1.1**

Version 1.2

April 2022

**Test Log**

1. ***Test Log Identifier***

Vortex-Website 1.2 release 1.1.TL2022

Note: the structure of this document is based on the IEEE 829-1998 Standard for Software Test Documentation.

2. ***Description***

The tests will be carried out as explained in the Test plan.

## F.2 Test Log

### F.2.1 Application Interface Test Log

| Test Name | Test Details | Validation | Results and Changes Made |
|---|---|---|---|
| React app launch | Testing the launch of the web User interface | Passed | The react application is running as supposed and even rendered on the cloud platform: heroku |
| Create account test | User is able to create new account with chosen informations, then stored in | Passed | A message on the console is printed with the new user information |
| Header redirection test | Testing the hypertext link from the header and see if it redirect correctly to chosen page | Passed | Redirection on features and technologies correctly functioning |
| Correctly importing background | Test if the animated background is correctly rendered | Passed | The background is correctly animated |
| Connection to MongoDB | Connection to database through user access key and password in .env | Passed | A message of successful connection is printed o the console |
| Auth feature test | Able to login through correct user information | Passed | Find existing user in the database and will redirect the user to the homepage |
| Auth feature test | Notification if password or email is incorrect | Passed | Message passing through alerts shown on the interface |
| Detection feature test | Correctly goes through algorithm and output vortex detection | Passed | Output of the edited image and cores informations displayed on the interface |
| Detection feature test | Alert when different format in input | Failed | As the input is supposed to be a picture if the user input other format the program won't work, but no alert is created |
| Tracking feature test | Correctly goes through algorithm and output vortex video | Failed | Currently not implemented as the AI team worked on a new environement, we still have issues adapting it to the previous one |
| Dashboard test | Able to show utilisation of features use from database | Failed | Currently the information is not rendered but shown in console |
| Integration Testing | A sequence of tests done to ensure the application tun correctly work when integration of each feature | Passed | Every feature implemented doesn't affect the others and the application is running smoothly |

Table 11: Testing Log Tables for the User interface

### F.2.2 YOLO Test Log

| Test Name | Test Details | Validation | Results and Changes Made |
|---|---|---|---|
| Build Custom Dataset | Check manually created labeled datasets for both models (YOLO and Faster RCNN) | Failed | Labeled image could not be applied equally to YOLO and Faster RCNN models Dataset was changed by using Roboflow. |
| Initial Training Test | (YOLO) Test made with custom dataset to check training session | Passed | Model worked, but it needed to improve accuracy. (Batch size: 32) |
| Initial Training Test (Faster RCNN) | Test made with custom dataset to check training session | Failed → Passed | The setting value(batch) could not keep same value with YOLO condition. (Batch size: 32 → 8) |
| Resizing Image test (416 → 640, YOLO) | Test made with changed resizing image size in training session | Passed | Model worked and it showed improved accuracy. |
| Resizing Image test (416 → 640, Faster RCNN) | Test made with changed resizing image size in training session | Failed → Passed | Model worked and it showed improved accuracy, but the batch size changed (Batch size: 8 → 4) |
| Reliability Check Test (YOLO v5 Small) | Test made with 3 types of datasets (100 / 300 / 700) to check reliability | Passed | Model worked and it showed the improvement following increasing data except 100 dataset (No. of Layers : 270) |
| Performance (accuracy) Check Test (YOLO v5 Medium) | Test made with 700 images dataset to check accuracy (mAP). Success was to show better performance than small model. | Passed | Model worked and it showed the improvement. (No. of Layers: 369) |
| Performance (accuracy) Check Test (YOLO v5 XLarge) | Test made with 700 images dataset to check accuracy (mAP). Success was to show better performance than small model. | Passed | Model worked and it showed the improvement. (No. of Layers: 567) |
| Performance (accuracy) Check Test (YOLO v5 Medium, 2,200 image samples) | Test made with 2,200 images dataset to check accuracy (mAP). Success was to show better performance than 700 images samples model. | Passed | Model worked and it showed the improvement following increasing data. |

Table 12: Testing Log Tables for the YOLO and CNN testing

### F.2.3 SIFT Neural Network Test Log

| Test Name | Test Details | Validation | Results and Changes Made |
|---|---|---|---|
| Validation of the Labelling | The manual labelling program was tested to ensure the output files were correct and compatible with both the SIFT model and the CNN models | Failed | YOLO takes the centre of the bounding box and the width information rather than the top left and bottom right point coordinates. The program was adjusted accordingly to create the correct outputs. |
| Initial Test | Test made with a small set of dummy data to make sure the learning model worked correctly with the specified arrays | Passed | - |
| Read files | Test checked whether the full set of files input were read properly | Passed | - |
| SIFT features test | Test to make sure the SIFT features were being generated properly for each image by printing out the values | Passed | - |
| Initial Training | Test with a small number of epochs to ensure the model training was working properly by testing with both training and validation data. | Failed | Model was achieving 100% accuracy. The image files trained on were improperly generated and had to be remade as a border was added to the images of vortices. |
| Epoch size tests | Training using different epoch numbers were performed to ensure the model was not overfitting to the dataset | Passed | Model was changed to plot the graphs of the metrics during the training process for validation purposes. |
| SIFT feature size tests | The model was trained using different feature sizes to ensure the program was working as intended | Failed | Some of the programs were not using the input feature sizes and had to be changed to make the training program more general. |
| Performance Tests | The model was tested for it's performance when changing the feature sizes and image size inputs. Success was considered to be that the model performed differently when using different parameters | Passed | The program had to be changed to ensure results of a sufficient detail were printed out. Both a classification matrix was drawn and a classification report was printed. |
| Validation of sliding window on images | The initial program was tested using a sliding window to classify regions through the SIFT model. | Failed | The program was not properly moving the sliding window so every region classified the same as the first region's classification. The program was adjusted properly to fix this. |
| Validation of the program's performance with different parameters | The program was tested using different sliding window dimensions and the corresponding models to test the performance of the classification method on the full images. | Passed | The method behaved as expected and the full results are explored in section 4.4.1. |

Table 13: Testing Log Tables for the SIFT Neural Network

### F.2.4 Core Finding Test Log

| Test Name | Test Details | Validation | Results and Changes Made |
|---|---|---|---|
| Initial Program Test | Tested the initial state of the program to ensure that it would successfully execute | Passed | - |
| Tests using different parameter values | The program was tested with a variety of different parameter values and the results were visualised to observe the impact of these values on the results | Failed | For the most part, this was successful as the programs would run as expected and would produce valid results. However, sometimes the curve_fit function would produce an error when it was not able to find the optimal gaussian. For this reason Error handling was implemented, outputting the centre of the bounding box if the program fails. |
| Data Sanity Tests | The outputs were visualised and observed to ensure that the results were being accurately produced | Failed | The Gaussian parameters that were found, tended to be as small as possible and most of the time did not fit the function at all. Further tests were done to find the optimal bounds and starting parameters. |
| Tests over assigned bounds | A sequence of tests done to test the optimal bounds to optimise the Gaussian Curve fitting parameters. | Passed | Bounds were successfully found which produced a realistic approximation to the curve rather than the poor approximations with the loose parameter bounds. |
| Integration Testing | A sequence of tests done to ensure the program and functions work properly when integrated with the main application | Passed | A few minor issues were had in the image formats needed in the function inputs, but the program succeeded and the results were combined and displayed in the front-end application. |

Table 14: Testing Log Tables for the Core Location Algorithm

# G Project Management Reports

## G.1 Scope Changes Log

This report captures all the changes made to the scope of work during the project

Report: Project Sprint_v1

*Issue added after sprint start

Scope changes log

View in issue navigator

| Date | Key | Summary | Issue type | Epic | Details of scope change | Change in estimation |
|---|---|---|---|---|---|---|
| 2022-03-09 | VOR-37* | SE-Code repository configuration | Story | CREATE BUILD PIPE... | Issue added to sprint | - |
| 2022-03-09 | VOR-38* | SE-CI server configuration | Story | CREATE BUILD PIPE... | Issue added to sprint | - |
| 2022-03-09 | VOR-37 | SE-Code repository configuration | Story | CREATE BUILD PIPE... | Estimate of 6 has been added | - → 6 |
| 2022-03-09 | VOR-38 | SE-CI server configuration | Story | CREATE BUILD PIPE... | Issue removed from sprint | - |
| 2022-03-10 | VOR-38* | SE-CI server configuration | Story | CREATE BUILD PIPE... | Issue added to sprint | - |
| 2022-03-14 | VOR-53* | CFD-Simulation (in progress + issues) | Story | CREATE IMAGES WI... | Issue added to sprint | - |
| 2022-03-14 | VOR-53 | CFD-Simulation (in progress + issues) | Story | CREATE IMAGES WI... | Issue removed from sprint | - |
| 2022-03-15 | VOR-54* | AI-Deep Learning implementation | Story | TRAIN AI MODEL T... | Issue added to sprint | - |
| 2022-03-15 | VOR-55* | AI-Results will be tested for generating dataset | Task | | Issue added to sprint | - |
| 2022-03-15 | VOR-32* | SE-Software implementation | Story | | Issue added to sprint | - |
| 2022-03-15 | VOR-53* | CFD-Simulation (in progress + issues) | Story | CREATE IMAGES WI... | Issue added to sprint | - |
| 2022-03-15 | VOR-65* | CFD-RANS Model Simulation | Task | GOLF BALL MODEL... | Issue added to sprint | - |
| 2022-03-15 | VOR-66* | CFD-Trying new compact mesh | Task | | Issue added to sprint | - |
| 2022-03-16 | VOR-69* | CFD-K- omega model simulation | Task | | Issue added to sprint | - |

Figure 76: Scope Changes Log

## G.2 Cumulative Flow Diagram

This report shows the various status of the project issues over time. According to the report, the software is incomplete and clearly open for improvement and more development in the future.

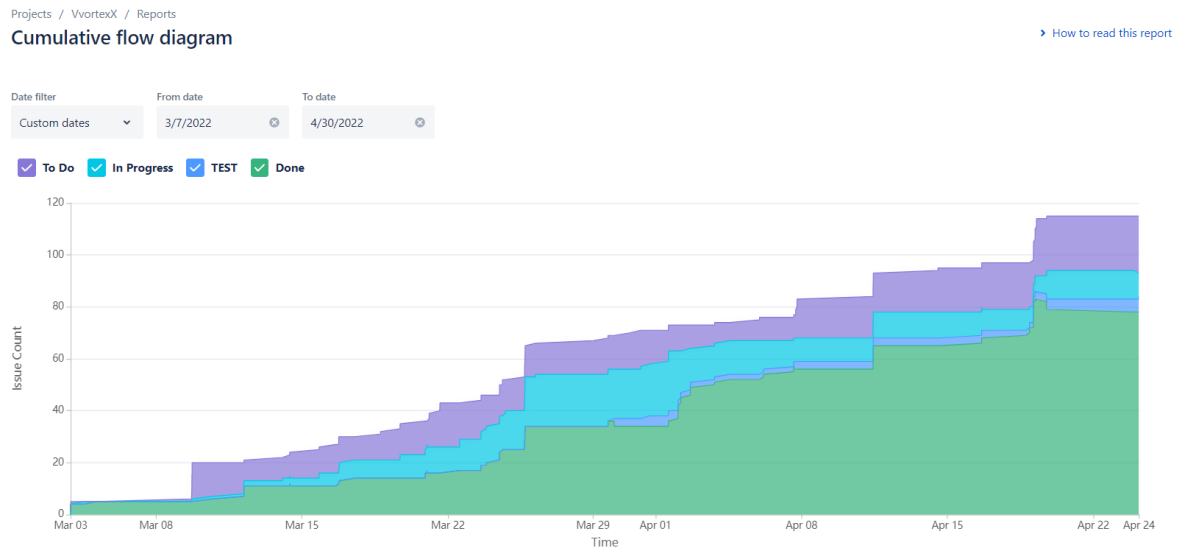



Figure 77: Cumulative Flow Diagram

# H Build Pipeline Report

The build pipeline was not successfully implemented. This will be further improved as stated in our future work.

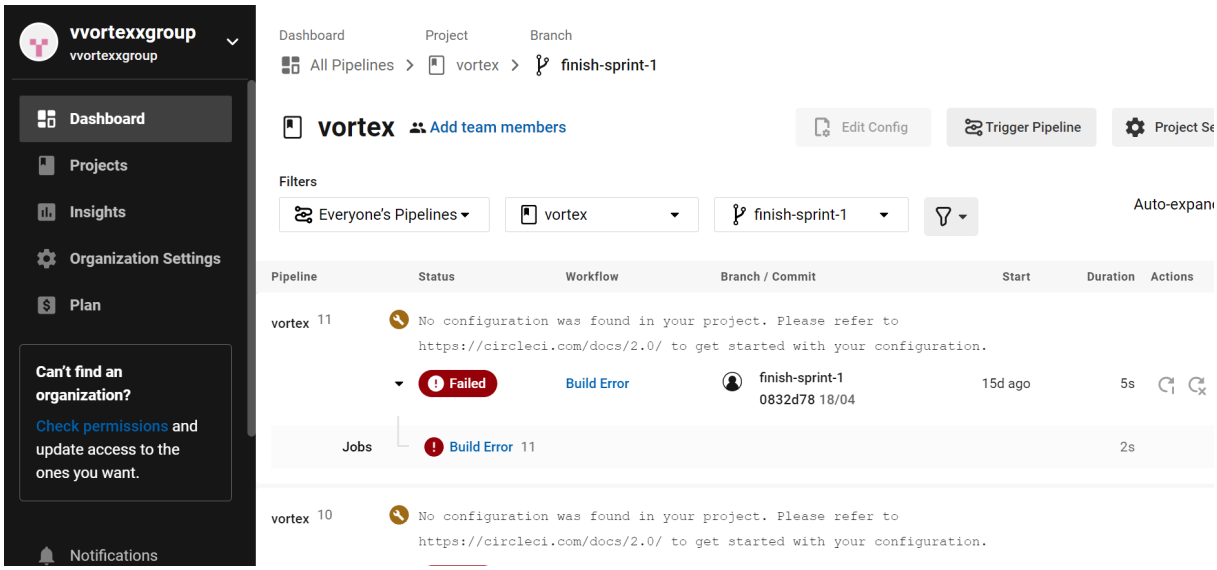



Figure 78: Build Pipeline

# I Pseudocode

**Algorithm 1** Core Finding Algorithm

**Input**
- F Image input pixel values
- n Iteration Count
- k Step Count
- c Update Rate

**Output**
- x x pixel location of core
- y y pixel location of core
- $\theta$ Orientation of the gaussian distribution
- $\sigma_x$ x sigma value of the gaussian distribution
- $\sigma_y$ y sigma value of the gaussian distribution

---

Calculate Sobel magnitude direction:
**for** $i < F.height$ **do**
  **for** $j < F.width$ **do**
    $sx[i][j] \leftarrow Sobel\_x(F)[i][j]$
    $sy[i][j] \leftarrow Sobel\_y(F)[i][j]$
    $sd[i][j] \leftarrow \tan^{-1}(\frac{sy[i][j]}{sx[i][j]})$
  **end for**
**end for**
Randomly vote for core location:
$V[i][j] \leftarrow 0$ for all $i \in [0, F.width - 1]$, $j \in [0, F.height - 1]$
**for** $i < n$ **do**
  $x \in Random(0, F.width - 1)$
  $y \in Random(0, F.height - 1)$
  **for** $j < k$ **do**
    $xi \leftarrow x + cos(sd[x][y]) * Random(0, c - 1) * Random([1, -1])$
    $yi \leftarrow y + sin(sd[x][y]) * Random(0, c - 1) * Random([1, -1])$
    $x \leftarrow xi$
    $y \leftarrow yi$
  **if** not (0 <= $x < F.width$ and 0 <= $y < F.height$) **then**
      Terminate Iteration and go back to next iteration of the 5: loop
  **end if**
    $V[x][y] \leftarrow V[x][y] + 1$
**end for**
Find optimal Parameters for Gaussian Distribution:
$x, y, \theta, \sigma_x, \sigma_y \leftarrow Optimise(Gaussian, V)$ using scipy to optimise =0

---

# J User Manual

| **Name** | **Usage** |
|---|---|
| Sign in button | User is able to create new account with chosen information in text fields |
| Home hypertext link | Redirection on Home page |
| Features hypertext link | Redirection on features section |
| Technologies hypertext link | Redirection on technologies section |
| Login button | User is able to authenticate himself |
| Detection button | Goes through algorithm and output vortex detection |
| Tracking button | Goes through algorithm and output vortex video |
| Dashboard button | Able to show utilisation of features use |
| Upload button | Allow user to upload an image/video |
| Save button | Save the insight and other information from a feature utilisation |

Table 15: User manual

# K Code Files

## K.1 Miscellaneous Scripts

### K.1.1 YOLO Training

Here is the script we used to train the YOLO model.

```
# -*- coding: utf-8 -*-
"""Yolov5m_2200s.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/19o-l7RlkO9tbYCPLShGGBLSyReQSOiOf

# YOLO v5 medium option Code for finding Vortics
"""

## Notice
## This code should be excuted on Google Colab
## Before running cell, we should change hardware accelerator in runtime type to 'GPU'

"""## Build Development Environment"""

## Mount google drive to secure data safety

from google.colab import drive
drive.mount('/content/gdrive')

## Create link replacement abbreviations

!ln -s /content/gdrive/My\ Drive/ /mydrive  #designate gdrive >> mydrive

# Commented out IPython magic to ensure Python compatibility.
## Generate Task folder

# %mkdir /content/gdrive/MyDrive/yolov5m_2200
# %cd /content/gdrive/MyDrive/yolov5m_2200

## Clone the YOLO v5 Model Repository


!git clone https://github.com/ultralytics/yolov5

# Commented out IPython magic to ensure Python compatibility.
## Upload training image files
```

```
## It is to be preferred to use Google Drive to upload training image files as it is too slow to upload in Colab.

# %cd /content/gdrive/MyDrive/yolov5m_2200/yolov5

from google.colab import files

#upload image data (images.zip)
uploaded = files.upload()

!ls -al

# Commented out IPython magic to ensure Python compatibility.
## Extract compressed image file

# %cd /content/gdrive/MyDrive/yolov5m_2200/yolov5

# %mkdir images
!unzip final_dataset.zip -d /content/gdrive/MyDrive/yolov5m_2200/yolov5/images

# Change nc(number of classes) in model file(yolo5s.yaml or...)

# Commented out IPython magic to ensure Python compatibility.
## Upload configuration yaml file

# %cd /content/gdrive/MyDrive/yolov5m_2200/yolov5/data

# upload vortex.yaml to data folder

from google.colab import files
uploaded = files.upload()
!ls -al

# Commented out IPython magic to ensure Python compatibility.
# Check vortex.yaml for correct path and configuration

# %cd /content/gdrive/MyDrive/yolov5m_2200/yolov5/data

# %cat vortex.yaml

## Import pytorch image processing function

import torch
from IPython.display import Image

print('torch %s %s' % (torch.__version__, torch.cuda.get_device_properties(0) if torch.cuda.is_available() else 'CPU'))
```

```
85
86 """## Train Model"""
87
88 # Commented out IPython magic to ensure Python compatibility.
89 ## Initiate model trainning
90
91 # %cd /content/gdrive/MyDrive/yolov5m_2200/yolov5
92
93 # img: define input image size
94 # batch: determine batch size
95 # epochs: define the number of training epochos
96 # data: set the path to our yaml file
97 # cfg: specify our model configuration
98 # weights: specify a custom path to weights (if you have)
99 # name: result names
100 # nosave: only save the final checkpoint
101 # cache: cache images for faster training
102
103 !python train.py --img 640 --batch 32 --data ↙
    ↪ "/content/gdrive/MyDrive/yolov5m_2200/yolov5/data/vortex.yaml" ↙
    ↪ --cfg ./models/yolov5m.yaml --weights ↙
    ↪ /content/gdrive/MyDrive/yolov5m_2200/yolov5/runs/ ↙
    ↪ train/yolov5m_2200_results12/weights/last.pt --name ↙
    ↪ yolov5m_2200_results --cache --epochs 1000
104
105 # Commented out IPython magic to ensure Python compatibility.
106 ## Generate validation data
107
108 # %cd /content/gdrive/MyDrive/yolov5m_2200/yolov5
109
110 !python val.py --weights ↙
    ↪ /content/gdrive/MyDrive/yolov5m_2200/yolov5/runs/train/ ↙
    ↪ yolov5m_2200_results12/weights/best.pt --data ↙
    ↪ /content/gdrive/MyDrive/yolov5m_2200/yolov5/data/vortex.yaml ↙
    ↪ --imgsz 640 --task val
111
112 """## Testing Model"""
113
114 # Commented out IPython magic to ensure Python compatibility.
115 ## Run detector on test image and Check elapsed time
116
117 import math
118 import time
119
120 start = time.time()
121 math.factorial(100000)
122
123 # %cd /content/gdrive/MyDrive/yolov5m_2200/yolov5
124
125 !python detect.py --weights ↙
```

```
    ↪ /content/gdrive/MyDrive/yolov5m_2200/yolov5/runs/train/ ↙
    ↪ yolov5m_2200_results12/weights/best.pt --conf 0.1 ↙
    ↪ --source ↙
    ↪ /content/gdrive/MyDrive/yolov5m_2200/yolov5/Plot_y_4th_data_o.mp4
126
127 end = time.time()
128
129 ## Print out Elapsed time
130 print(f"{end - start:.5f} sec")
131
132 # Commented out IPython magic to ensure Python compatibility.
133 ## Check the status on trained model using tensor board
134
135 # %load_ext tensorboard
136 # %tensorboard --logdir ↙
    ↪ '/content/gdrive/MyDrive/yolov5m_2200/yolov5/runs/train/ ↙
    ↪ yolov5m_2200_results12'
```

### K.1.2 Faster RCNN Training

Here is the script we used to train the RCNN model.

```
1  # -*- coding: utf-8 -*-
2  """Faster_RCNN_Vortics.ipynb
3
4  Automatically generated by Colaboratory.
5
6  Original file is located at
7      https://colab.research.google.com/drive/ ↙
           ↪ 1FMGzTC4j_PIdY_Pr5QJyr--mzln-GqKx
8  """
9
10
11
12 """# Faster R-CNN Code for finding Vortics"""
13
14 ## Notice
15 ## This code should be excuted on Google Colab
16 ## Before running cell, we should change hardware ↙
       ↪ accelerator in runtime type to 'GPU'
17
18 """## Build Development Environment """
19
20 ## Mount google drive to secure data safety
21
22 from google.colab import drive
23 drive.mount('/content/gdrive')
24
25 ## Create link replacement abbreviations
26
```

```
!ln -s /content/gdrive/MY\ Drive/ /mydrive

# Commented out IPython magic to ensure Python compatibility.
## Generate Task folder

# %mkdir /content/gdrive/MyDrive/f_rcnn
# %cd /content/gdrive/MyDrive/f_rcnn

## Clone the TensorFlow Models Repository including Faster ↙
    ↳ R-CNN

import os
import pathlib

if "models" in pathlib.Path.cwd().parts:
  while "models" in pathlib.Path.cwd().parts:
    os.chdir('..')
elif not pathlib.Path('models').exists():
  !git clone --depth 1 https://github.com/tensorflow/models

# Commented out IPython magic to ensure Python compatibility.
# ## Install the Object Detection API
#
# %%bash
# cd /content/gdrive/MyDrive/f_rcnn/models/research
# protoc object_detection/protos/*.proto --python_out=.
# cp object_detection/packages/tf2/setup.py .
# python -m pip install .

# Commented out IPython magic to ensure Python compatibility.
## Import functions from repository and public

import matplotlib
import matplotlib.pyplot as plt

import os
import random
import io
import imageio
import glob
import scipy.misc
import numpy as np
from six import BytesIO
from PIL import Image, ImageDraw, ImageFont
from IPython.display import display, Javascript
from IPython.display import Image as IPyImage

import tensorflow as tf

from object_detection.utils import label_map_util
```

```
from object_detection.utils import config_util
from object_detection.utils import visualization_utils as ↙
    ↳ viz_utils
from object_detection.utils import colab_utils
from object_detection.builders import model_builder

# %matplotlib inline

## Test Model Builder, temporarily

!python /content/gdrive/MyDrive/f_rcnn/models/research/ ↙
    ↳ object_detection/builders/model_builder_tf2_test.py

## Define loading function of image file into numpy array

def load_image_into_numpy_array(path):

  img_data = tf.io.gfile.GFile(path, 'rb').read()
  image = Image.open(BytesIO(img_data))
  (im_width, im_height) = image.size
  return np.array(image.getdata()).reshape(
      (im_height, im_width, 3)).astype(np.uint8)

## Define wrapper function to visualize detections

def plot_detections(image_np,
                    boxes,
                    classes,
                    scores,
                    category_index,
                    figsize=(12, 16),
                    image_name=None):

  image_np_with_annotations = image_np.copy()
  viz_utils.visualize_boxes_and_labels_on_image_array(
      image_np_with_annotations,
      boxes,
      classes,
      scores,
      category_index,
      use_normalized_coordinates=True,
      min_score_thresh=0.8)
  if image_name:
    plt.imsave(image_name, image_np_with_annotations)
  else:
    plt.imshow(image_np_with_annotations)

## Designate TFRecord(Image files dataset record) names and ↙
    ↳ link

```

```
test_record_fname = '/content/gdrive/MyDrive/f_rcnn/test/test.tfrecord'
train_record_fname = '/content/gdrive/MyDrive/f_rcnn/train/train.tfrecord'
label_map_pbtxt_fname = '/content/gdrive/MyDrive/f_rcnn/train/label_map.pbtxt'



## Remain option model and code to be able to choose different models available in the TF2 object detection zoo

MODELS_CONFIG = {
    'efficientdet-d0': {
        'model_name': 'efficientdet_d0_coco17_tpu-32',
        'base_pipeline_file': 'ssd_efficientdet_d0_512x512_coco17_tpu-8.config',
        'pretrained_checkpoint': 'efficientdet_d0_coco17_tpu-32.tar.gz',
        'batch_size': 16
    },
    'faster_rcnn_resnet101_v1': {
        'model_name': 'faster_rcnn_resnet101_v1_640x640_coco17_tpu-8',
        'base_pipeline_file': 'faster_rcnn_resnet101_v1_640x640_coco17_tpu-8.config',
        'pretrained_checkpoint': 'faster_rcnn_resnet101_v1_640x640_coco17_tpu-8.tar.gz',
        'batch_size': 4
    }
}

chosen_model = 'faster_rcnn_resnet101_v1'

num_steps = 40000
num_eval_steps = 500

model_name = MODELS_CONFIG[chosen_model]['model_name']
pretrained_checkpoint = MODELS_CONFIG[chosen_model]['pretrained_checkpoint']
base_pipeline_file = MODELS_CONFIG[chosen_model]['base_pipeline_file']
batch_size = MODELS_CONFIG[chosen_model]['batch_size'] # The default value is 4 for our task (Number of training images : 700)

# Commented out IPython magic to ensure Python compatibility.
## Download pretrained weights of resnet101

```

```
# %mkdir /content/gdrive/MyDrive/f_rcnn/models/research/deploy/
# %cd /content/gdrive/MyDrive/f_rcnn/models/research/deploy/
import tarfile
download_tar = 'http://download.tensorflow.org/models/ ↙
    ↳ object_detection/tf2/20200711/' + pretrained_checkpoint

!wget {download_tar}
tar = tarfile.open(pretrained_checkpoint)
tar.extractall()
tar.close()



# Commented out IPython magic to ensure Python compatibility.
## download base training configuration file of resnet101

# %cd /content/gdrive/MyDrive/f_rcnn/models/research/deploy
download_config = ↙
    ↳ 'https://raw.githubusercontent.com/tensorflow/models/ ↙
    ↳ master/research/object_detection/configs/tf2/' + ↙
    ↳ base_pipeline_file
!wget {download_config}

# Commented out IPython magic to ensure Python compatibility.
## Generate folder in order to upload training and testing ↙
    ↳ image record (TFrecord)

# %mkdir /content/gdrive/MyDrive/f_rcnn/test
# %mkdir /content/gdrive/MyDrive/f_rcnn/train

## Prepare training environment

pipeline_fname = ↙
    ↳ '/content/gdrive/MyDrive/f_rcnn/models/research/deploy/' ↙
    ↳ + base_pipeline_file
fine_tune_checkpoint = ↙
    ↳ '/content/gdrive/MyDrive/f_rcnn/models/research/deploy/' ↙
    ↳ + model_name + '/checkpoint/ckpt-0'

## Define function for collecting classes to train model

def get_num_classes(pbtxt_fname):
    from object_detection.utils import label_map_util
    label_map = label_map_util.load_labelmap(pbtxt_fname)
    categories = ↙
        ↳ label_map_util.convert_label_map_to_categories(
         label_map, max_num_classes=90, use_display_name=True)
    category_index = ↙
        ↳ label_map_util.create_category_index(categories)
    return len(category_index.keys())
```

```
num_classes = get_num_classes(label_map_pbtxt_fname)

# Commented out IPython magic to ensure Python compatibility.
## Write custom configuration file by slotting our dataset, ↙
    ↳ model checkpoint, and training parameters into the base ↙
    ↳ pipeline file

import re

# %cd /content/gdrive/MyDrive/f_rcnn/models/research/deploy
print('writing custom configuration file')

with open(pipeline_fname) as f:
    s = f.read()
with open('pipeline_file.config', 'w') as f:

    # fine_tune_checkpoint
    s = re.sub('fine_tune_checkpoint: ".*?"',
               'fine_tune_checkpoint: ↙
                  ↳ "{}"'.format(fine_tune_checkpoint), s)

    # tfrecord files train and test.
    s = re.sub(
        '(input_path: ↙
           ↳ ".*?)(PATH_TO_BE_CONFIGURED/train)(.*?")', ↙
           ↳ 'input_path: "{}"'.format(train_record_fname), s)
    s = re.sub(
        '(input_path: ↙
           ↳ ".*?)(PATH_TO_BE_CONFIGURED/val)(.*?")', ↙
           ↳ 'input_path: "{}"'.format(test_record_fname), s)

    # label_map_path
    s = re.sub(
        'label_map_path: ".*?"', 'label_map_path: ↙
           ↳ "{}"'.format(label_map_pbtxt_fname), s)

    # Set training batch_size.
    s = re.sub('batch_size: [0-9]+',
               'batch_size: {}'.format(batch_size), s)

    # Set training steps, num_steps
    s = re.sub('num_steps: [0-9]+',
               'num_steps: {}'.format(num_steps), s)

    # Set number of classes num_classes.
    s = re.sub('num_classes: [0-9]+',
               'num_classes: {}'.format(num_classes), s)

    #fine-tune checkpoint type
    s = re.sub(
```

```
240         'fine_tune_checkpoint_type: "classification"', ↙
              ↳ 'fine_tune_checkpoint_type: ↙
              ↳ "{}"'.format('detection'), s)
241
242     f.write(s)
243
244 # Commented out IPython magic to ensure Python compatibility.
245 # %cat ↙
    ↳ /content/gdrive/MyDrive/f_rcnn/models/research/deploy/ ↙
    ↳ pipeline_file.config
246
247 ## Set config file link and main path for training file
248
249 pipeline_file = ↙
    ↳ '/content/gdrive/MyDrive/f_rcnn/models/research/deploy/ ↙
    ↳ pipeline_file.config'
250 model_dir = '/content/gdrive/MyDrive/f_rcnn/training/'
251
252 ## Correct the version error of opencv in repository
253
254 !pip uninstall opencv-python-headless
255
256 ## Correct the version error of opencv in repository
257
258 !pip install opencv-python-headless==4.1.2.30
259
260
261
262 """## Train Model"""
263
264 ## Initiate model trainning
265
266 !python /content/gdrive/MyDrive/f_rcnn/models/research/ ↙
    ↳ object_detection/model_main_tf2.py \
267     --pipeline_config_path={pipeline_file} \
268     --model_dir={model_dir} \
269     --alsologtostderr \
270     --num_train_steps={num_steps} \
271     --sample_1_of_n_eval_examples=1 \
272     --num_eval_steps={num_eval_steps}
273
274 ## run model evaluation to obtain performance metrics
275 #!python ↙
    ↳ /content/models/research/object_detection/model_main_tf2.py ↙
    ↳ \
276     #--pipeline_config_path={pipeline_file} \
277     #--model_dir={model_dir} \
278     #--checkpoint_dir={model_dir}
279
280 # Commented out IPython magic to ensure Python compatibility.
```

```
## Check the status on trained model using tensor board

# %load_ext tensorboard
# %tensorboard --logdir '/content/gdrive/MyDrive/f_rcnn/training/train'



# Commented out IPython magic to ensure Python compatibility.
## Check the saved weights
# %ls '/content/gdrive/MyDrive/f_rcnn/training/'

## Run conversion script

import re
import numpy as np

output_directory = '/content/gdrive/MyDrive/f_rcnn/fine_tuned_model'


last_model_path = '/content/gdrive/MyDrive/f_rcnn/training/'
print(last_model_path)
!python /content/gdrive/MyDrive/f_rcnn/models/research/object_detection/exporter_main_v2.py \
    --trained_checkpoint_dir {last_model_path} \
    --output_directory {output_directory} \
    --pipeline_config_path {pipeline_file}

# Commented out IPython magic to ensure Python compatibility.
## Save generated model file

# %ls '/content/gdrive/MyDrive/f_rcnn/fine_tuned_model/saved_model/'



"""## Set testing environment"""

# Commented out IPython magic to ensure Python compatibility.
## Download test images with format COCO JSON to Google Drive

# %cd /content/gdrive/MyDrive/f_rcnn/test/
!unzip planex_re_10.zip -d /content/gdrive/MyDrive/f_rcnn/test3
#!curl -L "[YOUR LINK HERE]" > roboflow.zip; unzip roboflow.zip; rm roboflow.zip

# Commented out IPython magic to ensure Python compatibility.
## Load functions for testing (Re-load the overlapped function because of lost loaded condition while long
```

```
    ↪ training time)
325
326 import matplotlib
327 import matplotlib.pyplot as plt
328
329 import io
330 import scipy.misc
331 import numpy as np
332 from six import BytesIO
333 from PIL import Image, ImageDraw, ImageFont
334
335 import tensorflow as tf
336
337 from object_detection.utils import label_map_util
338 from object_detection.utils import config_util
339 from object_detection.utils import visualization_utils as ↙
    ↪ viz_utils
340 from object_detection.builders import model_builder
341
342 # %matplotlib inline
343
344 ## Define loading function of image file into numpy array ↙
    ↪ (Re-load the overlapped function because of lost loaded ↙
    ↪ condition while long training time)
345
346 def load_image_into_numpy_array(path):
347
348   img_data = tf.io.gfile.GFile(path, 'rb').read()
349   image = Image.open(BytesIO(img_data))
350   (im_width, im_height) = image.size
351   return np.array(image.getdata()).reshape(
352       (im_height, im_width, 3)).astype(np.uint8)
353
354 # Commented out IPython magic to ensure Python compatibility.
355 ## Check the saved weights
356
357 # %ls '/content/gdrive/MyDrive/f_rcnn/training/'
358
359 # Define testing environment and object detection function
360
361 import pathlib
362
363 filenames = ↙
    ↪ list(pathlib.Path('/content/gdrive/MyDrive/f_rcnn/ ↙
    ↪ training/').glob('*.index'))
364
365 filenames.sort()
366 print(filenames)
367
368 #recover our saved model
```

```
pipeline_config = pipeline_file
#generally you want to put the last ckpt from training in here
model_dir = str(filenames[-1]).replace('.index','')
configs = config_util.get_configs_from_pipeline_file(pipeline_config)
model_config = configs['model']
detection_model = model_builder.build(
      model_config=model_config, is_training=False)

# Restore checkpoint
ckpt = tf.compat.v2.train.Checkpoint(
      model=detection_model)
ckpt.restore(os.path.join(str(filenames[-1]).replace('.index','')))


def get_model_detection_function(model):
  """Get a tf.function for detection."""

  @tf.function
  def detect_fn(image):
    """Detect objects in image."""

    image, shapes = model.preprocess(image)
    prediction_dict = model.predict(image, shapes)
    detections = model.postprocess(prediction_dict, shapes)

    return detections, prediction_dict, tf.reshape(shapes, [-1])

  return detect_fn

detect_fn = get_model_detection_function(detection_model)

## Map labels for inference decoding

label_map_path = configs['eval_input_config'].label_map_path
label_map = label_map_util.load_labelmap(label_map_path)
categories = label_map_util.convert_label_map_to_categories(
    label_map,
    max_num_classes=label_map_util.get_max_label_map_index(label_map),
    use_display_name=True)
category_index = label_map_util.create_category_index(categories)
label_map_dict = label_map_util.get_label_map_dict(label_map, use_display_name=True)

## Run detector on test image (it takes a little longer on the first run and then runs at normal speed)

```

```
413 import random
414 import math
415 import time
416
417 # Remain random Choice function
418 # TEST_IMAGE_PATHS = glob.glob('/content/gdrive/MyDrive/f_rcnn/test3/*.jpg')
419 # image_path = random.choice(TEST_IMAGE_PATHS)
420
421 TEST_IMAGE_PATHS = glob.glob('/content/gdrive/MyDrive/f_rcnn/test3/*.png')
422
423 # Check the elapsed time
424 start = time.time()
425 math.factorial(100000)
426
427 #print(len(TEST_IMAGE_PATHS))
428 for i in range(len(TEST_IMAGE_PATHS)):
429
430   image_path = TEST_IMAGE_PATHS[i]
431   image_np = load_image_into_numpy_array(image_path)
432   i = i + 1
433
434 # Things to try:
435 # Flip horizontally
436 # image_np = np.fliplr(image_np).copy()
437
438 # Convert image to grayscale
439 # image_np = np.tile(
440 #     np.mean(image_np, 2, keepdims=True), (1, 1, 3)).astype(np.uint8)
441
442   input_tensor = tf.convert_to_tensor(
443       np.expand_dims(image_np, 0), dtype=tf.float32)
444   detections, predictions_dict, shapes = detect_fn(input_tensor)
445
446   label_id_offset = 1
447   image_np_with_detections = image_np.copy()
448
449   viz_utils.visualize_boxes_and_labels_on_image_array(
450         image_np_with_detections,
451         detections['detection_boxes'][0].numpy(),
452         (detections['detection_classes'][0].numpy() + label_id_offset).astype(int),
453         detections['detection_scores'][0].numpy(),
454         category_index,
455         use_normalized_coordinates=True,
456         max_boxes_to_draw=200,
457         min_score_thresh=.1,
```

```
458           agnostic_mode=False,
459     )
460
461     plt.figure(figsize=(12,16))
462 #   fig=plt.imshow(image_np_with_detections)
463 #   plt.close(fig)
464 #plt.show()
465
466 ## Save the result images
467
468     plt.imsave('/content/gdrive/MyDrive/f_rcnn/results/a' + str(i) + '.png',image_np_with_detections)
469
470 end = time.time()
471
472 ## Print out Elapsed time
473
474 print(f"{end - start:.5f} sec")
```

### K.1.3 Vortex Image Labelling Program

Here is the script we used to label the vortices in our dataset.

```
1  """
2  This File uses code from the OpenCV Open Source Library. The use of this library is acknowledged here and
3  a link is left to the OpenCV homepage:
4  https://opencv.org/
5  """
6
7  import cv2 as cv
8  from os import listdir
9
10 ix, iy = -1,-1
11
12 # Defines SIFT and image parameters
13 sift_box_size = 16
14 sift_box_count = 4
15 sub_image_size = sift_box_size * sift_box_count
16 sub_image_count = 10
17 sub_image_deviation = 24
18
19
20 # Defines the call-back function to check for the click events
21 def draw_rect(event, x, y, flags, param):
22     global ix, iy
23     # Checks for when the user presses down the left mouse button
24     if event == cv.EVENT_LBUTTONDOWN:
25         cv.circle(img, (x, y), 2, (0, 255, 0), -1)
```

```
26          ix, iy = x, y
27      # Checks for when the user lifts up the left mouse ↙
            ↳ button and appends the selected rectangle
28      elif event == cv.EVENT_LBUTTONUP:
29          cv.rectangle(img, (ix, iy), (x, y), (0, 255, 0), 1)
30          box_positions.append([(ix, iy), (x, y)])
31
32
33  # Defines the directory paths to read from
34  image_paths = [
35      "final_images/images1",
36      "final_images/images2",
37      "final_images/images2b",
38      "final_images/images4",
39      "final_images/images-2k/plane1",
40      "final_images/images-2k/plane2",
41      "final_images/images-2k/plane3",
42      "final_images/images-2k-smooth/planex",
43      "final_images/images-2k-smooth/planey",
44      "final_images/images-2k-smooth/planez",
45      "final_images/images-2k-smooth/planezz",
46      "final_images/images-3k/x",
47      "final_images/images-3k/y",
48      "final_images/images-3k/z",
49      "final_images/images-4k/planex",
50      "final_images/images-4k/planey",
51      "final_images/images-4k/planez",
52      "final_images/images-3k-smooth/planex",
53      "final_images/images-3k-smooth/planey",
54      "final_images/images-3k-smooth/planez",
55      "final_images/images-3k-smooth/planezz",
56  ]
57
58  # Creates a list of the image pathways
59  names = [listdir(path) for path in image_paths]
60  names = [[image_paths[j] + "/" + names[j][i] for i in ↙
        ↳ range(len(names[j]))] for j in range(len(image_paths))]
61  files = [f for paths in names for f in paths]
62  boxes_list = [[] for _ in range(len(files))]
63
64  # Defines the OpenCV window and call-back function
65  cv.namedWindow('image')
66  cv.setMouseCallback('image', draw_rect)
67
68  # Iterates over each image in the list
69  i = 0
70  while i < len(boxes_list):
71      print(str(i+1) + "/" + str(len(boxes_list)))
72
73      # Reads in the image
```

```
74      f = files[i]
75      img = cv.imread(f)
76      img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
77
78      # Takes a copy of the image in case the user undoes a box
79      img_original = img.copy()
80      box_positions = boxes_list[i].copy()
81      for box in box_positions:
82          cv.circle(img, box[0], 2, (0, 255, 0), -1)
83          cv.rectangle(img, box[0], box[1], (0, 255, 0), 1)
84
85      # Defines parameters for loop
86      not_exit = True
87      go_back = False
88      while not_exit:
89          cv.imshow('image', img)
90
91          # Gets current key pressed
92          k = cv.waitKey(1) & 0xFF
93
94          # If d is pressed, move onto the next image
95          if k == ord('d'):
96              not_exit = False
97
98          # If s is pressed, remove the last box made
99          elif k == ord('s'):
100             img = img_original.copy()
101             box_positions = box_positions[:-1]
102             # Redraws the remaining boxes
103             for box in box_positions:
104                 cv.circle(img, box[0], 2, (0, 255, 0), -1)
105                 cv.rectangle(img, box[0], box[1], (0, 255, ↙
                        ↪ 0), 1)
106
107         # if a is pressed, go back to the last image
108         elif k == ord('a'):
109             go_back = True
110             not_exit = False
111
112     # Checks whether we are going to the next image or the ↙
            ↪ previous image
113     boxes_list[i] = box_positions
114     if go_back and i != 0:
115         i -= 1
116     elif go_back and i == 0:
117         i = i
118     else:
119         i += 1
120
121 cv.destroyAllWindows()
```

```
122
123 # Writes each box to the output text files
124 for i in range(len(files)):
125     # Reads the image to get the height and width
126     f = files[i]
127     img = cv.imread(f)
128     height, width, _ = img.shape
129     boxes = boxes_list[i]
130
131     # Creates the output text file for each box
132     with open("final_vortex_locations_2/" + ↙
          ↳ f.replace(".png", ".txt"), "w") as file:
133         # Writes an entry for each vortex box labelled
134         for box in boxes:
135             file.write("0 " + str(((box[0][0] + box[1][0]) / ↙
                  ↳ 2) / width) + " " +
136                        str(((box[0][1] + box[1][1]) / 2) / ↙
                           ↳ height) + " " +
137                        str(abs(box[0][0] - box[1][0]) / ↙
                           ↳ width) + " " +
138                        str(abs(box[0][1] - box[1][1]) / ↙
                           ↳ height))
139             file.write('\n')
```

### K.1.4 Core Labelling Program

Here is the script we used to label the vortex cores for each labelled vortex.

```
1 """
2 This File uses code from the OpenCV Open Source Library. The ↙
     ↳ use of this library is acknowledged here and
3 a link is left to the OpenCV homepage:
4 https://opencv.org/
5 """
6
7 import cv2 as cv
8 from os import listdir
9
10 mouseX = 0
11 mouseY = 0
12
13 # Call-back function to set the mouse click location
14 def get_click(event, x, y, flags, param):
15     global mouseX, mouseY
16     # Checks for left button click condition
17     if event == cv.EVENT_LBUTTONUP:
18         # Changes image colour at location and changes the ↙
              ↳ global variables
19         cv.circle(subimg, (x, y), 3, 255, -1)
20         mouseX, mouseY = x, y
```

```
21
22
23 # Reads in every file from the selected files in 'image_paths'
24 image_paths = ["final_images/images1", ↙
    ↳ "final_images/images2", "final_images/images2b", ↙
    ↳ "final_images/images4"]
25 names = [listdir(path) for path in image_paths]
26 names = [[image_paths[j] + "/" + names[j][i] for i in ↙
    ↳ range(len(names[j]))] for j in range(len(image_paths))]
27 files = [f for paths in names for f in paths]
28
29 # Defines the OpenCV window with the call-back function
30 windowName = "ImageBox"
31 cv.namedWindow(windowName)
32 cv.setMouseCallback(windowName, get_click)
33
34 # Begins loop to iterate over each file
35 i = 0
36 while i < len(files):
37     print(str(i + 1) + "/" + str(len(files)))
38
39     # Reads in the image from path
40     f = files[i]
41     img = cv.imread(f)
42     img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
43     height, width = img.shape
44
45     # Opens the vortex location and creates a .txt file to ↙
        ↳ write core location to
46     with open(f.replace("final_images", ↙
        ↳ "final_vortex_locations_2").replace(".png", ↙
        ↳ ".txt"), "r") as in_file:
47         with open(f.replace("final_images", ↙
            ↳ "vortex_cores").replace(".png", ".txt"), "w") ↙
            ↳ as out_file:
48             # Reads in each vortex location and iterates ↙
                ↳ over each one
49             lines = in_file.readlines()
50             for line in lines:
51                 # Reads the vortex coordinates
52                 coords = line.split()
53                 x1 = round((float(coords[1]) - ↙
                    ↳ float(coords[3]) / 2) * width)
54                 x2 = round((float(coords[1]) + ↙
                    ↳ float(coords[3]) / 2) * width)
55                 y1 = round((float(coords[2]) - ↙
                    ↳ float(coords[4]) / 2) * height)
56                 y2 = round((float(coords[2]) + ↙
                    ↳ float(coords[4]) / 2) * height)
57
```

```
58                  # Creates a box around the vortex in the full image
59                  cv.rectangle(img, (x1, y1), (x2, y2), 255)
60                  cv.imshow("image2", img)
61
62                  # Gets the subimage of the vortex and enlarges it to display to the user
63                  subimg = img[y1:y2, x1:x2]
64                  subimg = cv.resize(subimg, (300, 300))
65
66                  # Keeps displaying the image until the user moves on
67                  keepProcessing = True
68                  while keepProcessing:
69                      # Waits for the user to press spacebar to indicate they have selected the centre
70                      cv.imshow(windowName, subimg)
71                      key = cv.waitKey(30)
72                      if key == ord(' '):
73                          keepProcessing = False
74                  cv.circle(img, (centerx, centery), 1, 255, -1)
75
76                  # Uses the selected coordinates to write to the output file
77                  centerx = round(mouseX * (x2 - x1) / 300) + x1
78                  centery = round(mouseY * (y2 - y1) / 300) + y1
79                  cv.circle(img, (centerx, centery), 1, 255, -1)
80                  out_file.write("0 " + str(centerx / width) + " " + str(centery / height) + "\n")
81      i += 1
```

### K.1.5 SIFT image generating program

Here is the script we used to generate the collection of small images used to train the SIFT model.

```
1   """
2   This File uses code from the OpenCV Open Source Library. The use of this library is acknowledged here and
3   a link is left to the OpenCV homepage:
4   https://opencv.org/
5   """
6
7   import cv2 as cv
8   import random
9   from os import listdir
10
11  ix, iy = -1, -1
12
```

```
13 # Defines the function parameters
14 sift_box_size = 16
15 sift_box_count = 2
16 sub_image_size = sift_box_size * sift_box_count
17 sub_image_count = 8
18 sub_image_deviation = 8
19 non_vortex_image_gap = sub_image_size
20 count_vortex = 1
21 count_nvortex = 1
22
23
24 # Defines the function to generate the sub images for the vortices
25 def create_sub_images(boxes):
26     # Gets the globabl variable for the vortex count for naming purposes
27     global count_vortex
28
29     # Defines the necessary border size so we don't deviate off the image
30     border_size = sub_image_deviation + sift_box_size * sift_box_count
31     img_border = cv.copyMakeBorder(
32         img,
33         top=border_size,
34         bottom=border_size,
35         left=border_size,
36         right=border_size,
37         borderType=cv.BORDER_CONSTANT,
38         value=0
39     )
40     half_sub_image = round(sub_image_size / 2)
41     height, width = img.shape
42
43     # Iterate over each bounding box in the input
44     for box in boxes:
45         # Gets the bounding box parameters
46         x = box[0][0]
47         y = box[0][1]
48         xi = box[1][0]
49         yi = box[1][1]
50         midX = round((x + xi) / 2)
51         midY = round((y + yi) / 2)
52
53         # Generates a number of sub images according to defined parameter
54         for i in range(sub_image_count):
55                 # Generates a random position
56                 randX = random.randint(-sub_image_deviation, sub_image_deviation)
```

```
57                  randY = random.randint(-sub_image_deviation, ↙
                        ↳ sub_image_deviation)
58
59                  # Generates the sub image corners for that ↙
                        ↳ random position
60                  x_1 = midX + randX - half_sub_image + ↙
                        ↳ border_size
61                  x_2 = midX + randX + half_sub_image + ↙
                        ↳ border_size
62                  y_1 = midY + randY - half_sub_image + ↙
                        ↳ border_size
63                  y_2 = midY + randY + half_sub_image + ↙
                        ↳ border_size
64
65                  # Gets and writes the sub image
66                  subImg = img_border[y_1:y_2, x_1:x_2]
67                  cv.imwrite("out_sub_images3/vortex/" + ↙
                        ↳ str(count_vortex) + ".png", subImg)
68
69                  # Increases the counter for naming purposes
70                  count_vortex += 1
71
72
73  # Defines the function to generate the sub images for the ↙
        ↳ non-vortex images
74  def create_non_sub_image(boxes):
75      # Gets the current global count
76      global count_nvortex
77
78      # Defines the initial sliding window region
79      x1 = 0
80      y1 = 0
81      x2 = sub_image_size
82      y2 = sub_image_size
83      height, width = img.shape
84
85      # Slides the window over the image until we reach the end
86      while y2 < height:
87          while x2 < width:
88              # For each window, checks whether it overlaps ↙
                    ↳ with one of the vortex bounding boxes
89              no_vortex = True
90              for box in boxes:
91                  if not (box[0][0] > x2 or box[1][0] < x1) ↙
                        ↳ and not (box[0][1] > y2 or box[1][1] < ↙
                        ↳ y1):
92                      no_vortex = False
93
94              # Writes the iamge to the dataset if there is no ↙
                    ↳ overlapping vortex boundign box
```

```
95              if no_vortex:
96                  subImg = img[y1:y2, x1:x2]
97                  cv.imwrite("out_sub_images3/non-vortex/" + ↙
                        ↳ str(count_nvortex) + ".png", subImg)
98                  count_nvortex += 1
99
100             # Increases the widnow locations to slide the ↙
                    ↳ window
101             x1 += non_vortex_image_gap
102             x2 += non_vortex_image_gap
103         x1 = 0
104         x2 = sub_image_size
105
106         y1 += non_vortex_image_gap
107         y2 += non_vortex_image_gap
108
109 # Defines the selected directories to obtain the images from
110 image_paths = ["final_images/images1", ↙
    ↳ "final_images/images2", "final_images/images2b", ↙
    ↳ "final_images/images4"]
111
112 # Obtains every path within these files to iterate through
113 names = [listdir(path) for path in image_paths]
114 names = [[image_paths[j] + "/" + names[j][i] for i in ↙
    ↳ range(len(names[j]))] for j in range(len(image_paths))]
115 files = [f for paths in names for f in paths]
116
117 # Iterates over the total list of vortices
118 i = 0
119 while i < len(files):
120     f = files[i]
121     # Reads the image in
122     img = cv.imread(f, 3)
123     img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
124     height, width = img.shape
125
126     # Reads in the labelled vortex locations
127     box_positions = []
128     with open(f.replace("final_images", ↙
        ↳ "final_vortex_locations_2/").replace(".png", ↙
        ↳ ".txt"), "r") as file:
129         lines = file.readlines()
130         for line in lines:
131             coords = line.split()
132             # Gets the bounding box for each vortex
133             x = round((float(coords[1]) - ↙
                    ↳ float(coords[3])/2) * width)
134             y = round((float(coords[2]) - ↙
                    ↳ float(coords[4])/2) * height)
135             x2 = round((float(coords[1]) + ↙
```

```
                ↪ float(coords[3])/2) * width)
136             y2 = round((float(coords[2]) + ↙
                ↪ float(coords[4])/2) * height)
137             box_positions.append(((x, y), (x2, y2)))
138
139     # Calls the functions to write the files
140     create_sub_images(box_positions)
141     create_non_sub_image(box_positions)
142     i += 1
```

### K.1.6 Train SIFT neural network

Here is the script which trained the SIFT neural network model and produced the figures.

```
1   """
2   This File uses code from the OpenCV Open Source Library. The ↙
        ↪ use of this library is acknowledged here and
3   a link is left to the OpenCV homepage:
4   https://opencv.org/
5
6   This File uses code from the TensorFlow Open Source Library. ↙
        ↪ The use of this library is acknowledged here and
7   a link is left to the TensorFlow homepage:
8   https://www.tensorflow.org/
9   """
10
11  import cv2 as cv
12  import numpy as np
13  import tensorflow as tf
14  import seaborn as sns
15  import random
16  from os import listdir
17  from matplotlib import pyplot as plt
18  from sklearn.metrics import classification_report, ↙
        ↪ confusion_matrix
19
20
21  # Define Parameters
22  sift_box_size = 16
23  sift_box_count = 2
24
25  sobelSize = 3
26
27
28  # Generates the SIFT feature descriptor for the input image ↙
        ↪ given the defined parameters
29  def generate_sift(img):
30      # Calculate the Sobel x and Sobel y responses to compute ↙
            ↪ the direction and magnitude
31      sobelx = cv.Sobel(img, cv.CV_64F, 1, 0, ksize=sobelSize)
```

```
32      sobely = cv.Sobel(img, cv.CV_64F, 0, 1, ksize=sobelSize)
33      sobel_mag = np.sqrt(np.square(sobelx) + np.square(sobely))
34      sobel_dir = cv.phase(sobelx, sobely, angleInDegrees=True)
35
36      # Assigns each directional response to the corresponding ↙
            ↳ histogram box
37      box = np.int8(np.floor(sobel_dir / 45))
38
39      # Defines the initial values for the SIFT feature
40      sift = np.array([[0.0 for _ in range(8)] for __ in ↙
            ↳ range(sift_box_count * sift_box_count)])
41
42      # Iterates over each point and adds the magnitude ↙
            ↳ response to the corresponding histogram
43      for i in range(sift_box_count):
44          for j in range(sift_box_count):
45              for xi in range(sift_box_size):
46                  for yj in range(sift_box_size):
47                      # Obtains the current pixel location
48                      xii = (i * sift_box_size) + xi
49                      yjj = (j * sift_box_size) + yj
50
51                      # Adds the magnitude value to the ↙
                            ↳ determined histogram
52                      sift[i * sift_box_count + ↙
                            ↳ j][box[yjj][xii]] += ↙
                            ↳ sobel_mag[yjj][xii]
53
54              # Normalises each histogram such that the values ↙
                    ↳ lie between 1.0 and 0.0
55              if np.linalg.norm(sift[i * sift_box_count + j]) ↙
                    ↳ != 0:
56                  sift[i * sift_box_count + j] = sift[i * ↙
                        ↳ sift_box_count + j] / np.sum(sift[i * ↙
                        ↳ sift_box_count + j])
57
58      # Flattens the histogram for use in the model
59      return sift.flatten()
60
61
62  # Obtains the list of vortex images and shuffles them
63  vortex_images = listdir("out_sub_images2/vortex")
64  vortex_images = ["out_sub_images2/vortex/" + path for path ↙
        ↳ in vortex_images]
65  random.shuffle(vortex_images)
66
67  # Obtains the list of non-vortex images and shuffles them
68  nvortex_images = listdir("out_sub_images2/non-vortex")
69  nvortex_images = ["out_sub_images2/non-vortex/" + path for ↙
        ↳ path in nvortex_images]
```

```
70 random.shuffle(nvortex_images)
71
72 # Defines the data split
73 train_test_split = 0.7
74
75 # Defines the number of vortices and non-vortices within ↙
     ↳ each dataset
76 vortex_cutoff = round(len(vortex_images) * train_test_split)
77 nvortex_cutoff = round(len(nvortex_images) * train_test_split)
78
79 # Gets the list of training files
80 train_paths = vortex_images[:vortex_cutoff] + ↙
     ↳ nvortex_images[:nvortex_cutoff]
81 random.shuffle(train_paths)
82
83 # Gets the list of testing files
84 test_paths = vortex_images[vortex_cutoff:] + ↙
     ↳ nvortex_images[nvortex_cutoff:]
85 random.shuffle(test_paths)
86
87 # Defines the training data and the testing data
88 sift_training = [[] for _ in range(vortex_cutoff + ↙
     ↳ nvortex_cutoff)]
89 labels_training = [[0, 0] for _ in range(vortex_cutoff + ↙
     ↳ nvortex_cutoff)]
90
91 sift_testing = [[] for _ in range(len(vortex_images) - ↙
     ↳ vortex_cutoff + len(nvortex_images) - nvortex_cutoff)]
92 labels_testing = [[0, 0] for _ in range(len(vortex_images) - ↙
     ↳ vortex_cutoff + len(nvortex_images) - nvortex_cutoff)]
93
94 # Reads in the training data, computes the SIFT feature and ↙
     ↳ writes it all to a text file
95 with open("train_64x64_2x2_32.txt", "w") as train:
96     for i in range(len(train_paths)):
97         # Prints the current file number we are on
98         print("train: " + str(i+1) + "/" + ↙
               ↳ str(len(train_paths)))
99
100        # Gets the image file
101        f = train_paths[i]
102        img = cv.imread(f)
103
104        # Computes the SIFT feature and adds it to the dataset
105        sift = generate_sift(img)
106        sift_training[i] = sift
107
108        # Writes the file to the output file
109        for j in range(sift_box_count * sift_box_count * 8):
110            train.write(str(sift[j]) + " ")
```

```
111
112         # Writes the classification values to the training ↙
                ↳ data and output data file
113         if "non-vortex" in f:
114             labels_training[i][0] = 0
115             labels_training[i][1] = 1
116             train.write("0 1\n")
117         else:
118             labels_training[i][0] = 1
119             labels_training[i][1] = 0
120             train.write("1 0\n")
121
122 # Reads in the testing data, computes the SIFT feature and ↙
        ↳ writes it all to a text file
123 with open("test_64x64_2x2_32.txt", "w") as test:
124     for i in range(len(test_paths)):
125         # Prints the current file number we are on
126         print("test: " + str(i + 1) + "/" + ↙
                ↳ str(len(test_paths)))
127
128         # Gets the image file
129         f = test_paths[i]
130         img = cv.imread(f)
131
132         # Computes the SIFT feature and adds it to the dataset
133         sift = generate_sift(img)
134         sift_testing[i] = sift
135
136         # Writes the file to the output file
137         for j in range(sift_box_count * sift_box_count * 8):
138             test.write(str(sift[j]) + " ")
139
140         # Writes the classification values to the testing ↙
                ↳ data and output data file
141         if "non-vortex" in f:
142             labels_testing[i][0] = 0
143             labels_testing[i][1] = 1
144             test.write("0 1\n")
145         else:
146             labels_testing[i][0] = 1
147             labels_testing[i][1] = 0
148             test.write("1 0\n")
149
150
151 # Rather than continuously having to wait for the data to be ↙
        ↳ generated,
152 # instead this commented-out block of code reads the data in ↙
        ↳ from the
153 # files we've written to
154 """with open("train_64x64_2x2_32.txt", "r") as train:
```

```
    lines = train.readlines()
    for i in range(len(lines)):
        print(str(i+1) + "/" + str(len(lines)))
        vals = lines[i].split()
        sift = [float(vals[i]) for i in range(sift_box_count ↙
            ↳ * sift_box_count * 8)]
        sift_training[i] = sift
        labels_training[i][0] = int(vals[sift_box_count * ↙
            ↳ sift_box_count * 8])
        labels_training[i][1] = int(vals[sift_box_count * ↙
            ↳ sift_box_count * 8 + 1])


with open("test_64x64_2x2_32.txt", "r") as test:
    lines = test.readlines()
    for i in range(len(lines)):
        print(str(i+1) + "/" + str(len(lines)))
        vals = lines[i].split()
        sift = [float(vals[i]) for i in range(sift_box_count ↙
            ↳ * sift_box_count * 8)]
        sift_testing[i] = sift
        labels_testing[i][0] = int(vals[sift_box_count * ↙
            ↳ sift_box_count * 8])
        labels_testing[i][1] = int(vals[sift_box_count * ↙
            ↳ sift_box_count * 8 + 1])"""


# Converts the data arguments to tensor objects
X = tf.constant(sift_training, dtype=tf.float16)
Y = tf.constant(labels_training, dtype=tf.float16)
X2 = tf.constant(sift_testing, dtype=tf.float16)
Y2 = tf.constant(labels_testing, dtype=tf.float16)

# Set up model
model = tf.keras.Sequential()

# Adds the layers to the model based on the specified ↙
    ↳ parameters
model.add(tf.keras.layers.Dense(64,
                                input_dim=sift_box_count * ↙
                                    ↳ sift_box_count * 8,
                                activation='relu'))

model.add(tf.keras.layers.Dense(32,
                                activation='relu'))

model.add(tf.keras.layers.Dense(2,
                                activation='softmax'))

# Defines the optimiser, loss value to minimise and the ↙
```

```
    ↪ metrics we are measuring
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy', ↙
                 ↪ tf.keras.metrics.Precision(), ↙
                 ↪ tf.keras.metrics.Recall()])

# Trains the Model
history = model.fit(
    X, ↙
       ↪ # Input training data
    Y, ↙
       ↪ # Output training data
    epochs=500,
    batch_size=200,
    verbose=1, ↙
       ↪ # Amount of detail you want shown in terminal while ↙
       ↪ training
    validation_data=(X2, Y2)
)

# Draws the metric graphs over the 500 epochs
plt.plot(range(500), history.history['val_accuracy'])
plt.xlabel("Epochs"), plt.ylabel("Accuracy")
plt.title('Accuracy')
plt.show()

plt.plot(range(500), history.history['val_precision'])
plt.xlabel("Epochs"), plt.ylabel("Precision")
plt.title('Precision')
plt.show()

plt.plot(range(500), history.history['val_recall'])
plt.xlabel("Epochs"), plt.ylabel("Recall")
plt.title('Recall')
plt.show()

# Saves the model
model.save('Model_64x64_2x2_32')

# Evaluates the model using the testing data
results = model.evaluate(X2, Y2)
Y2 = Y2.numpy()
Y3 = model(X2).numpy()
print("Results of testing are: ", results)

# Computes the classification report and the confusion matrix
a = np.argmax(Y2, axis=1)
b = np.argmax(Y3, axis=1)

```

```
r = classification_report(b, a, digits=10, output_dict=True)
c = confusion_matrix(b, a)

# Reorders the confusion matrix so the figure appears correct
f = c[1][1]
c[1][1] = c[1][0]
c[1][0] = f
d = np.sum(c, axis=1)
d = [[d[0]], [d[1]]]
c = c / d

# Draws the heatmap based on the confusion matrix
plt.figure(figsize=(2, 2))
heat_map = sns.heatmap(c, linewidths=1, annot=True)
plt.xticks([0.5, 1.5], ["Positives", "Negative"])
plt.yticks([0.5, 1.5], ["Vortex", "Non-Vortex"], va="center")
plt.title("Classification Matrix")
plt.show()

# Prints the classification report
print(classification_report(b, a, digits=10))
```

### K.1.7 Test SIFT neural network

Here is the script used to test the neural network using a sliding window.

```
"""
This File uses code from the OpenCV Open Source Library. The use of this library is acknowledged here and
a link is left to the OpenCV homepage:
https://opencv.org/

This File uses code from the TensorFlow Open Source Library. The use of this library is acknowledged here and
a link is left to the TensorFlow homepage:
https://www.tensorflow.org/
"""

import cv2 as cv
import numpy as np
import tensorflow as tf
from os import listdir

# Defines the function parameters
sift_box_size = 32
sift_box_count = 2
sub_image_size = sift_box_size * sift_box_count
image_gap = 32
sobelSize = 3

```

```
23
24 # Generates the SIFT feature descriptor for the input image ↙
   ↳ given the defined parameters
25 def generate_sift(img):
26     # Calculate the Sobel x and Sobel y responses to compute ↙
       ↳ the direction and magnitude
27     sobelx = cv.Sobel(img, cv.CV_64F, 1, 0, ksize=sobelSize)
28     sobely = cv.Sobel(img, cv.CV_64F, 0, 1, ksize=sobelSize)
29     sobel_mag = np.sqrt(np.square(sobelx) + np.square(sobely))
30     sobel_dir = cv.phase(sobelx, sobely, angleInDegrees=True)
31
32     # Assigns each directional response to the corresponding ↙
       ↳ histogram box
33     box = np.int8(np.floor(sobel_dir / 45))
34
35     # Defines the initial values for the SIFT feature
36     sift = np.array([[0.0 for _ in range(8)] for __ in ↙
       ↳ range(sift_box_count * sift_box_count)])
37
38     # Iterates over each point and adds the magnitude ↙
       ↳ response to the corresponding histogram
39     for i in range(sift_box_count):
40         for j in range(sift_box_count):
41             for xi in range(sift_box_size):
42                 for yj in range(sift_box_size):
43                     # Obtains the current pixel location
44                     xii = (i * sift_box_size) + xi
45                     yjj = (j * sift_box_size) + yj
46
47                     # Adds the magnitude value to the ↙
                       ↳ determined histogram
48                     sift[i * sift_box_count + ↙
                       ↳ j][box[yjj][xii]] += ↙
                       ↳ sobel_mag[yjj][xii]
49
50             # Normalises each histogram such that the values ↙
               ↳ lie between 1.0 and 0.0
51             if np.linalg.norm(sift[i * sift_box_count + j]) ↙
               ↳ != 0:
52                 sift[i * sift_box_count + j] = sift[i * ↙
                   ↳ sift_box_count + j] / np.sum(sift[i * ↙
                   ↳ sift_box_count + j])
53
54     # Flattens the histogram for use in the model
55     return sift.flatten()
56
57
58 # Function determines whether two rectangles overlap
59 def overlap(box1, box2):
60     # Checks if one box is on the left of the other
```

```
61      if box1[0][0] >= box2[1][0] or box2[0][0] >= box1[1][0]:
62          return False
63      # Checks if one box is below the other
64      elif box1[1][1] <= box2[0][1] or box2[1][1] <= box1[0][1]:
65          return False
66      # Return true otherwise
67      return True
68
69
70  # Loads in the model indicated in the string
71  model = tf.keras.models.load_model("Model_64x64_2x2_32")
72
73  # Defines the selected directories to obtain the test images ↙
        ↳ from
74  image_paths = ["final_images/images1", "final_images/images2"]
75
76  # Obtains every path within these files to iterate through
77  names = [listdir(path) for path in image_paths]
78  names = [[image_paths[j] + "/" + names[j][i] for i in ↙
        ↳ range(len(names[j]))] for j in range(len(image_paths))]
79  files = [f for paths in names for f in paths]
80
81  # Iterates over each input file
82  for i in range(len(files)):
83      # Prints the current image number
84      print(str(i+1) + "/" + str(len(files)))
85
86      # Reads the image
87      f = files[i]
88      img = cv.imread(f)
89      img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
90      height, width = img.shape
91
92      # Reads the true vortex locations
93      vortex_boxes = []
94      with open(f.replace(".png", ↙
            ↳ ".txt").replace("final_images", ↙
            ↳ "final_vortex_locations_2")) as in_file:
95          lines = in_file.readlines()
96          for line in lines:
97              coords = line.split()
98              # Reads the true vortex locations and appends ↙
                    ↳ them to vortex_boes
99              x1 = round((float(coords[1]) - ↙
                    ↳ float(coords[3])/2) * width)
100             x2 = round((float(coords[1]) + ↙
                    ↳ float(coords[3])/2) * width)
101             y1 = round((float(coords[2]) - ↙
                    ↳ float(coords[4])/2) * height)
102             y2 = round((float(coords[2]) + ↙
```

```
                ↪ float(coords[4])/2) * height)
103             vortex_boxes.append([(x1, y1), (x2, y2)])
104
105     # Makes a copy of the image for drawing the boxes
106     imgcopy = img.copy()
107
108     # Draws the true vortex locations on the displayed image
109     for box in vortex_boxes:
110         cv.rectangle(imgcopy, box[0], box[1], 0)
111
112     # Defines the initial positions of the sliding window
113     x1 = 0
114     y1 = 0
115     x2 = sub_image_size
116     y2 = sub_image_size
117     height, width = img.shape
118
119     # Contains the SIFT feature responses for each window
120     sifts = []
121
122     # Moves the sliding window and calculates the SIFT ↙
        ↪ feature response
123     while y2 < height:
124         while x2 < width:
125             # Gets the SIFT feature
126             sift_out = generate_sift(img[y1:y2, x1:x2])
127             sifts.append(sift_out)
128
129             # Moves the sliding window
130             x1 += image_gap
131             x2 += image_gap
132         x1 = 0
133         x2 = sub_image_size
134         y1 += image_gap
135         y2 += image_gap
136
137     # Determines the Model's classification for each sliding ↙
        ↪ window location
138     x = tf.constant(sifts, dtype=tf.float16)
139     prediction = model(x)
140
141     # Defines the initial positions of the sliding window
142     x1 = 0
143     y1 = 0
144     x2 = sub_image_size
145     y2 = sub_image_size
146     j = 0
147
148     # Moves the sliding window to draw the predicted regions
149     while y2 < height:
```

```
150             while x2 < width:
151                 # Gets the model's predictions
152                 pred = prediction[j]
153 
154                 # Labels the region as a vortex if it is the ↙
                        ↳ most likely
155                 # possibility and is greater than the assigned ↙
                        ↳ threshold
156                 if pred[0] > pred[1] and pred[0] > 0.7:
157                     # Draws the outline of the region
158                     cv.rectangle(imgcopy, (x1, y1), (x2, y2), 255)
159 
160                 # Slides the window along
161                 x1 += image_gap
162                 x2 += image_gap
163                 j += 1
164             x1 = 0
165             x2 = sub_image_size
166             y1 += image_gap
167             y2 += image_gap
168 
169         # Uses thresholding to get the outlines of each larger ↙
                ↳ region
170         _, thresh = cv.threshold(imgcopy, 254, 255, ↙
                ↳ cv.THRESH_BINARY)
171         img_regions = img.copy()
172         cv.imshow("Responses", imgcopy)
173 
174         # Finds the contours in the threshold to get the ↙
                ↳ outermost bounding boxes
175         contours, hierarchy = cv.findContours(thresh, ↙
                ↳ cv.RETR_EXTERNAL, cv.CHAIN_APPROX_NONE)
176 
177         # For each contour get the bounding box and draw it
178         for contour in contours:
179             # Gets the bounding box
180             x, y, w, h = cv.boundingRect(contour)
181 
182             # Draws the bounding box
183             cv.rectangle(img_regions, (x, y), (x+w, y+h), 255)
184 
185         # Shows the resulting boxes
186         cv.imshow("Regions", img_regions)
187 
188         # Waits for a key press before displaying the next image
189         cv.waitKey(0)
```

### K.1.8 Core Finder Evaluating program

Here is the script used to evaluate the Core finding method and compare them to the true locations and the naive centre of the bounding box approach.

```
"""
This File uses code from the OpenCV Open Source Library. The use of this library is acknowledged here and
a link is left to the OpenCV homepage:
https://opencv.org/
"""

import cv2 as cv
import numpy as np
import math
import scipy.optimize as opt
from os import listdir

# Defines the function parameters
sobelSize = 3
update_rate = 3
sample_size = 10000
step_count = 100
added_region = 0


# Declares the Gaussian that the optimisation step is optimising over
def twoD_Gaussian(xy, amplitude, xo, yo, sigma_x, sigma_y, theta, offset):
    x, y = xy
    x = np.array(x)
    y = np.array(y)
    xo = float(xo)
    yo = float(yo)
    a = (np.cos(theta) ** 2) / (2 * sigma_x ** 2) + (np.sin(theta) ** 2) / (2 * sigma_y ** 2)
    b = -(np.sin(2 * theta)) / (4 * sigma_x ** 2) + (np.sin(2 * theta)) / (4 * sigma_y ** 2)
    c = (np.sin(theta) ** 2) / (2 * sigma_x ** 2) + (np.cos(theta) ** 2) / (2 * sigma_y ** 2)
    g = offset + amplitude * np.exp(-(a * ((x - xo) ** 2) + 2 * b * (x - xo) * (y - yo) + c * ((y - yo) ** 2)))
    return g.ravel()


# Applies the gradient descent algorithm to the vortex image
def gradient_descent(vortex_img):
    # Computes the Sobel x and Sobel y operators to determine the edge directions within the image
    sobelx = cv.Sobel(vortex_img, cv.CV_64F, 1, 0,
```

```
        ↪ ksize=sobelSize)
39      sobely = cv.Sobel(vortex_img, cv.CV_64F, 0, 1, ↙
        ↪ ksize=sobelSize)
40      sobel_dir = np.array(cv.phase(sobelx, sobely, ↙
        ↪ angleInDegrees=True))
41
42      # Copies the vortex image's dimensions to create the ↙
        ↪ probability distribution
43      placement_img = vortex_img.copy()
44      placement_img[:, :] = 0
45
46      height, width = vortex_img.shape
47
48      # Randomly allocates the random positions for the entire ↙
        ↪ sample size
49      x_currents = np.random.randint(0, width-1, sample_size)
50      y_currents = np.random.randint(0, height-1, sample_size)
51
52      # Iterates for the specified step count
53      for _ in range(step_count):
54          # Gets tje directional responses for each location
55          dirs = sobel_dir[y_currents, x_currents]
56
57          # Moves each sample a random amount in the direction ↙
            ↪ specified by it's sobel response
58          directions = np.random.choice([-1, 1], len(dirs))
59          x_currents += np.int32(np.round(directions * ↙
            ↪ np.cos(np.radians(dirs)) * ↙
            ↪ np.random.random(len(x_currents)) * update_rate))
60          y_currents += np.int32(np.round(directions * ↙
            ↪ np.sin(np.radians(dirs)) * ↙
            ↪ np.random.random(len(y_currents)) * update_rate))
61
62          # Zips the objects together to determine any samples ↙
            ↪ which have gone 'out-of-bounds'
63          xyzip = np.array(list(zip(x_currents, y_currents)))
64
65          # Terminates the process if there are no remaining ↙
            ↪ samples. Instead we return the centre of the ↙
            ↪ bounding box
66          if len(xyzip) == 0:
67              popt = [0.9, round(width / 2), round(height / ↙
                ↪ 2), 1.5, 1.5, 0, 0.1]
68              return popt
69          # Removes samples which have gone 'out-of-bounds'
70          xyzip = xyzip[(0 <= xyzip[:, 0]) & (xyzip[:, 0] < ↙
            ↪ width) & (0 <= xyzip[:, 1]) & (xyzip[:, 1] < ↙
            ↪ height)]
71
72          # Unzips the locations
```

```
73          x_currents = xyzip[:, 0]
74          y_currents = xyzip[:, 1]
75
76      # Gets the number of each entry and the number of votes ↙
            ↳ at that location
77      idx, cnt = np.unique(list(zip(x_currents, y_currents)), ↙
            ↳ return_counts=True, axis=0)
78
79      # Assigns the values to the probability distribution
80      for i in range(len(idx)):
81          placement_img[idx[i][1], idx[i][0]] += cnt[i]
82
83      # Votes tended to favour the edges of the image so ↙
            ↳ remove the votes on the image edges.
84      placement_img[:, 0] = 0
85      placement_img[:, width-1] = 0
86      placement_img[0, :] = 0
87      placement_img[height-1, :] = 0
88
89      # Blur the distribution to smooth it
90      placement_img = cv.GaussianBlur(placement_img, (3,3), 1)
91
92      # Divides by the max such that the values lie between ↙
            ↳ 1.0 and 0.0
93      placement_img = placement_img / np.max(placement_img)
94
95      # Adds an empty border to the image to allow for more ↙
            ↳ complex gaussian approximations
96      border_size = 10
97      placement_img = cv.copyMakeBorder(
98          placement_img,
99          top=border_size,
100         bottom=border_size,
101         left=border_size,
102         right=border_size,
103         borderType=cv.BORDER_CONSTANT,
104         value=0
105     )
106
107     # Gets the probability responses for each location in ↙
            ↳ the distribution
108     xy_in = ([x for x, y in ↙
            ↳ np.ndindex((placement_img.shape[1], ↙
            ↳ placement_img.shape[0]))],
109                 [y for x, y in ↙
                        ↳ np.ndindex((placement_img.shape[1], ↙
                        ↳ placement_img.shape[0]))])
110     probs = [placement_img[y, x] for x, y in ↙
            ↳ np.ndindex((placement_img.shape[1], ↙
            ↳ placement_img.shape[0]))]
```

```
111
112     # Define our initial gaussian prediction
113     initial_pred = (0.95, round(placement_img.shape[1]/2), ↙
            ↳ round(placement_img.shape[0]/2), 3, 3, 0, 0.1)
114     try:
115         # Attempts to find the optimal distribution ↙
                ↳ according to the specified parameters
116         popt, pcov = opt.curve_fit(twoD_Gaussian, xy_in, ↙
                ↳ probs, p0=initial_pred,
117                                    bounds=([0.9, border_size, ↙
                                           ↳ border_size, 1.0, 1.0, ↙
                                           ↳ 0, 0],
118                                                [2, ↙
                                                   ↳ placement_img.shape[1] ↙
                                                   ↳ - border_size, ↙
                                                   ↳ placement_img.shape[0] ↙
                                                   ↳ - border_size, ↙
                                                   ↳ placement_img.shape[1], ↙
                                                   ↳ placement_img.shape[0], ↙
                                                   ↳ 2 * np.pi, 0.3]))
119     # If an error is encountered we return the bounding box ↙
            ↳ centre
120     except RuntimeError:
121         popt = [0.9, round(width/2), round(height/2), 1.5, ↙
                ↳ 1.5, 0, 0.1]
122         return popt
123     except ValueError:
124         popt = [0.9, round(width/2), round(height/2), 1.5, ↙
                ↳ 1.5, 0, 0.1]
125         return popt
126
127     # Reduce the centre locations so they line up with the ↙
            ↳ image without the border
128     popt[1] = popt[1] - border_size
129     popt[2] = popt[2] - border_size
130     return popt
131
132
133 # Draws the Gaussian Distribution as defined in the input ↙
        ↳ parameters
134 def draw_gauss(popt, vortex_img):
135     # Gets all the x and y locations in the image
136     xy_in = ([x for x, y in np.ndindex((vortex_img.shape[1], ↙
            ↳ vortex_img.shape[0]))],
137                [y for x, y in np.ndindex((vortex_img.shape[1], ↙
                    ↳ vortex_img.shape[0]))])
138
139     # Gets the probability response for every location in ↙
            ↳ the image
140     fitted = twoD_Gaussian(xy_in, *popt)
```

```
141
142     # Draws the probabilities on a blank iamge
143     placement_img3 = np.float32(vortex_img.copy())
144     for i in range(len(fitted)):
145         placement_img3[xy_in[1][i], xy_in[0][i]] = fitted[i]
146
147     # Resizes the image such that the arrowed lines can be ↙
        ↪ drawn to represent the orientations
148     placement_img3 = cv.resize(placement_img3, ↙
        ↪ (vortex_img.shape[1] * 5, vortex_img.shape[0] * 5))
149     placement_img3 = cv.cvtColor(np.float32(placement_img3), ↙
        ↪ cv.COLOR_GRAY2BGR)
150     cv.arrowedLine(placement_img3,
151                    (round(popt[1] * 5 + 1), round(popt[2] * ↙
                       ↪ 5 + 1)),
152                    (round((popt[1] + (np.cos(popt[5]) * ↙
                       ↪ popt[3] * 2)) * 5 + 1),
153                     round((popt[2] - (np.sin(popt[5]) * ↙
                        ↪ popt[3] * 2)) * 5 + 1)),
154                    (0, 150, 0),
155                    1)
156
157     cv.arrowedLine(placement_img3,
158                    (round(popt[1] * 5 + 1), round(popt[2] * ↙
                       ↪ 5 + 1)),
159                    (round((popt[1] + (np.cos(popt[5] + ↙
                       ↪ math.pi / 2) * popt[4] * 2)) * 5 + 1),
160                     round((popt[2] - (np.sin(popt[5] + ↙
                        ↪ math.pi / 2) * popt[4] * 2)) * 5 + ↙
                        ↪ 1)),
161                    (0, 150, 0),
162                    1)
163     # Draws the centre of the gaussian distribution
164     cv.circle(placement_img3, (round(popt[1] * 5 + 1), ↙
        ↪ round(popt[2] * 5 + 1)), 2, (150, 0, 0), -1)
165     return placement_img3
166
167
168 # Defines the image directories to iterate over
169 image_paths = ["final_images/images1", ↙
    ↪ "final_images/images2", "final_images/images4"]
170
171 # Gets the file paths for each image
172 names = [listdir(path) for path in image_paths]
173 names = [[image_paths[j] + "/" + names[j][i] for i in ↙
    ↪ range(len(names[j]))] for j in range(len(image_paths))]
174 files = [f for paths in names for f in paths]
175
176 # Gets the corresponding list of vortex cores and bounding ↙
    ↪ boxes
```

```
177 files_cores = []
178 for i in range(len(files)):
179     files_cores.append(files[i].replace("final_images", ↙
        ↳ "vortex_cores").replace(".png", ".txt"))
180 files_boxes = []
181 for i in range(len(files)):
182     files_boxes.append(files[i].replace("final_images", ↙
        ↳ "final_vortex_locations_2").replace(".png", ".txt"))
183
184 # Iterates over the list of image files
185 for i in range(len(files)):
186     # Prints the current file number
187     print(str(i+1) + "/" + str(len(files)))
188
189     # Reads the current image in
190     img = cv.imread(files[i])
191     img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
192     height, width = img.shape
193
194     # Gets the vortex bounding box for the image
195     vortex_boxes = []
196     with open(files_boxes[i]) as box_file:
197         with open(files_cores[i]) as core_file:
198             # Reads the list of labelled cores and the list ↙
                ↳ of labelled vortices
199             core_lines = core_file.readlines()
200             box_lines = box_file.readlines()
201
202             # Iterates over each vortex and gets the vortex ↙
                ↳ and core locations for each labelled vortex
203             for j in range(len(core_lines)):
204                 # Gets the core coordinates
205                 core_coords = core_lines[j].split()
206                 core_x = round(float(core_coords[0]) * width)
207                 core_y = round(float(core_coords[1]) * height)
208                 coords = box_lines[j].split()
209
210                 # Gets the bounding box coordinates
211                 x1 = round((float(coords[1]) - ↙
                    ↳ float(coords[3])/2) * width)
212                 x2 = round((float(coords[1]) + ↙
                    ↳ float(coords[3])/2) * width)
213                 y1 = round((float(coords[2]) - ↙
                    ↳ float(coords[4])/2) * height)
214                 y2 = round((float(coords[2]) + ↙
                    ↳ float(coords[4])/2) * height)
215
216                 # Appends the vortex box to the list
217                 vortex_boxes.append([(x1, y1), (x2, y2)])
218
```

```
                    # Gets the gaussian prediction for each bounding box
                    popt = gradient_descent(img[max(y1-added_region, 0):min(y2+added_region, height-1), max(x1-added_region, 0):min(x2+added_region, width-1)])

                    # Plots the gaussian function as an image with orientation arrows indicated
                    gauss = draw_gauss(popt, img[max(y1-added_region, 0):min(y2+added_region, height-1), max(x1-added_region, 0):min(x2+added_region, width-1)])

                    # Displays the image
                    cv.imshow("gauss", gauss)
                    cv.waitKey(0)

                    # Calculates the difference between the prediction and the true labelled core location
                    x_pred = popt[1] + max(x1-added_region, 0)
                    y_pred = popt[2] + max(y1-added_region, 0)
                    print("Gradient Descent gives: " + str(math.sqrt((core_x - x_pred) ** 2 + (core_y - y_pred) ** 2)))

                    # Calculates the difference between the centre of the bounding box and the labelled core location
                    x_center = round((x1 + x2) / 2)
                    y_center = round((y1 + y2) / 2)
                    print("Center gives: " + str(math.sqrt((core_x - x_center) ** 2 + (core_y - y_center) ** 2)))
```

## K.2 React application files

dashboard.js:

```
import React, { Component} from 'react';
import { Button } from 'react-bootstrap';
import axios from 'axios';
class Dashboard extends Component {
  state = {
    use: []
  }
```

```
8
9   loadUtilisations = () => {
10    axios.post('http://localhost:3001/api/dashboard', {
11      method: 'POST',
12      headers: {
13        'Content-Type': 'application/json',
14        Authorization: `Token ${this.props.token}`
15      },
16      body: JSON.stringify({
17        insight:this.state.insight,
18        username:this.state.username,
19        date:this.state.date
20      })
21    })
22    .then(response => {
23      {/*const data = response.data;
24      this.setState({ use: data });*/}
25      console.log('Data has been received!!');
26      console.log(response.data)
27    })
28    .catch(() => {
29        alert('Error retrieving data!!!');
30      });
31  }
32  displayUSE = (data) => {
33
34      return data.map((data, index) => (
35        <div key={index} >
36          <h3>{data.username}</h3>
37          <p>{data.insight}</p>
38          <p>{data.date}</p>
39        </div>
40      ));
41
42    };
43
44  render() {
45    console.log('State: ', this.state);
46    return (
47      <div>
48      <button onClick={this.loadUtilisations}>Load
          ↳ history</button>
49        <h1>Use history</h1>
50         {this.displayUSE(this.state.use)}
51      </div>
52    );
53  }
54 }
55
56 export default Dashboard;
```

detection.js:

```
import React, { Component } from "react";


import axios from 'axios';
import './css element/Detection.css';


class Detection extends Component {
  constructor(props) {
     super(props);

     this.state = {
       imageInput: '',
       imageOutput: ''
     };

     this.handleUploadImage = ↙
        ↳ this.handleUploadImage.bind(this);
     this.onSubmit = this.onSubmit.bind(this);
     this.changeUsername = this.changeUsername.bind(this)
   }

   handleUploadImage(ev) {
     ev.preventDefault();
     const data = new FormData();

     data.append('image', ev.target.files[0]);
     this.setState({
      file: URL.createObjectURL(ev.target.files[0])
    })
     axios.all([
        axios.post("http://localhost:5000/detectObject", data)
        .then((res) => {

            this.setState({base64File: res.data.pic1});
            this.setState({base64File1: res.data.pic2});
            this.setState({info: res.data.data1});
            {/* this.setState({pic: res.data});*/}

              console.log(res.data.data1);
                {/*console.log(res.data.pic2);*/}
            {/* //console.log(this.state.pic);*/}
        })
        .catch(err => console.warn(err)),

    ])


   }
```

```
49
50      onSubmit(event){
51        event.preventDefault()
52        const use= {
53          insight:this.state.info,
54          username:this.state.username
55
56        }
57        axios.post ('http://localhost:3001/api/use',use)
58          .then(response => {
59            console.log(response.data)
60          })
61
62      }
63
64      changeUsername(event){
65        this.setState({
66           username:event.target.value
67        })
68      }
69
70
71      render() {
72        return (
73          <form >
74             <div class="file-upload">
75              <div class="image-upload-wrap">
76                 <input type="file" ↙
                       ↳ onChange={this.handleUploadImage}/>
77                 <div class="file-upload-content">
78                  <img alt=" " src={this.state.file}/>
79                 </div>
80              </div>
81            <img alt=" " ↙
                 ↳ src={`${this.state.base64File}`} ↙
                 ↳ />
82            <img alt=" " ↙
                 ↳ src={`${this.state.base64File1}`} ↙
                 ↳ />
83            {/* <img alt=" " ↙
                 ↳ src={`${this.state.pic}`} ↙
                 ↳ />*/}
84            <input type='text'
85            placeholder='username'
86            onChange={this.changeUsername}
87            value={this.state.username}/>
88            {this.state.info}
89           <button onClick={this.onSubmit}>Save the insights ↙
                 ↳ </button>
90          </div>
```

```
      </form>
      );
    }

}

export default Detection;
```

Features.js:

```
import React, { Component } from "react";
import Fade from "react-reveal";
import {BrowserRouter,Routes,Route,Link} from "react-router-dom";
import {Button} from "react-bootstrap";
import './css element/Features.css';
import Detection from "./detection";
import Tracking from './tracking';
import Dashboard from './dashboard';

class Features extends Component {
  render() {
    return (
      <section id="features">
        <Fade duration={1000}>
          <h2 className="title">Features :</h2>
          <div className="Switch1">

            <div className="Switch2">

              <Button className="btn"><Link to="/detection" >Detection</Link></Button>
              <Routes>
                <Route path="/detection" element={<Detection />} />
              </Routes>

              <Button className="btn"><Link to="/tracking" >Tracking</Link>  </Button>
              <Routes>
                <Route path="/tracking" element={<Tracking />} />
              </Routes>

              <Button className="btn"><Link to="/dashboard" >Dashboard</Link>  </Button>
              <Routes>
                <Route path="/dashboard" element={<Dashboard/>} />
              </Routes>

```

```
            </div>
          </div>
        </Fade>
      </section>
    );
  }
}

export default Features;
```

Footer.js:

```
import React, { Component } from "react";
import Fade from "react-reveal";

class Footer extends Component {
  render() {
    if (!this.props.data) return null;

    const networks = this.props.data.social.map(function (network) {
      return (
        <li key={network.name}>
          <a href={network.url}>
            <i className={network.className}></i>
          </a>
        </li>
      );
    });

    return (
      <footer>
        <div className="row">
          <Fade bottom>
            <div className="twelve columns">
              <ul className="social-links">{networks}</ul>

              <ul className="copyright">

                <li>
                  Design by{" "}
                  <a title="Styleshout" href="http://www.styleshout.com/">
                    Styleshout
                  </a>
                </li>
                <li>
                  Template by{" "}
                  <a title="Styleshout" href="https://github.com/nordicgiant2">
                    NordicGiant2
```

```
                  </a>
                </li>
                <li>
                  Edited by {" "}
                  <a title="Styleshout" ↙
                     ↳ href="https://whispering-harbor-38933.herokuapp.co
                    VvortexX
                  </a>
                </li>
              </ul>
            </div>
          </Fade>

          <div id="go-top">
            <a className="smoothscroll" title="Back to Top" ↙
               ↳ href="#home">
              <i className="icon-up-open"></i>
            </a>
          </div>
        </div>
      </footer>
    );
  }
}

export default Footer;
```

Header.js:

```
import React, { Component } from "react";
import {BrowserRouter,Routes,Route,Link} from ↙
   ↳ "react-router-dom";
import {Button} from "react-bootstrap";
import ParticlesBg from "particles-bg";
import Fade from "react-reveal";
//import ReactDOM from "react-dom";
import Login from "./Login";

import axios from 'axios';
import './css element/Header.css';
import logo from './css element/vortexlogo.png';

class Header extends Component {
  constructor(props) {
    super(props);
    this.state = {
      login: true
    };
    this.changeState = this.changeState.bind(this);
  }
  changeState = () => {
```

```
22      this.setState({
23        login: !this.state.login
24      });
25    };
26
27    render() {
28      if (!this.props.data) return null;
29      const name = this.props.data.name;
30      const { login } = this.state;
31      return (
32
33        <div class="hero">
34          <nav >
35            <img src={logo} width="300" height="100" />
36            <ul >
37              <li className="current">
38                <a className="smoothscroll" href="#home">Home</a>
39              </li>
40
41              <li>
42                <a className="smoothscroll" ↙
                     ↳ href="#features">Features</a>
43              </li>
44
45              <li>
46                <a className="smoothscroll" ↙
                     ↳ href="#technologies">Technologies</a>
47              </li>
48              <li>
49                <a><Link to="/login" > {login ? " Login" : " ↙
                     ↳ Log Out"}</Link>  </a>
50                <Routes>
51                  <Route path="/login" element={<Login />} />
52                </Routes>
53              </li>
54            </ul>
55          </nav>
56
57          <h1 className="responsive-headline">{name}</h1>
58          <a className="smoothscroll" href="#features">
59          <button>Check out our features &#8594;</button>
60          </a>
61
62
63        </div>
64
65
66
67      );
68    }
```

```
}
```

```
export default Header;
```

Login.js:

```
import React, { Component } from "react";
import axios from 'axios';
import Signup from "./SignUp";
import {BrowserRouter,Routes,Route,Link} from "react-router-dom";
import {Button} from "react-bootstrap";

class Login extends Component {
  constructor(){
    super()
    this.state= {
        username:'',
        email:'',
        password:''
    };
    this.changeEmail = this.changeEmail.bind(this)
    this.changePassword = this.changePassword.bind(this)
    this.loginUser = this.loginUser.bind(this);
  }


  changePassword(event){
    this.setState({
        password:event.target.value
    })
  }
  changeEmail(event){
    this.setState({
        email:event.target.value
    })
  }
  async loginUser(event) {
                event.preventDefault()

                const response = await fetch('http://localhost:3001/api/login', {
                    method: 'POST',
```

```
36                          headers: {
37                                  'Content-Type': ↙
                                      ↳ 'application/json',
38                          },
39                          body: JSON.stringify({
40          email:this.state.email,
41          password:this.state.password
42                          }),
43                  })
44
45                  const data = await response.json()
46
47                  if (data.user) {
48                          localStorage.setItem('token', ↙
                                ↳ data.user)
49                          alert('Login successful')
50                          window.location.href = '/'
51                  } else {
52                          alert('Please check your email and ↙
                                ↳ password')
53                  }
54          }
55
56    render() {
57      return(
58        <div>
59          <div className='container'>
60            <div className='form-div'>
61              <form>
62                <input type='text'
63                placeholder='E-mail'
64                onChange={this.changeEmail}
65                value={this.state.email}
66                className='form-control form-group'
67                />
68
69                <input type='password'
70                placeholder='Password'
71                onChange={this.changePassword}
72                value={this.state.password}
73                className='form-control form-group'
74                />
75               <button onClick={this.loginUser}>login</button>
76            </form>
77
78            <p>don't have an account</p>
79            <Button><Link to="/signup" >Sign Up</Link></Button>
80            <Routes>
81              <Route path="/*" element={<Signup />} />
82            </Routes>
```

```
83
84          </div>
85        </div>
86      </div>
87    );
88    }
89  }
90
91  export default Login;
```

SignUp.js:

```
1   import React, { Component } from "react";
2   import axios from 'axios';
3   //import 'bootstrap/dist/css/bootstrap.min.css';
4
5   class Signup extends Component {
6     constructor(){
7       super()
8       this.state= {
9           username:'',
10          email:'',
11          password:''
12      };
13      this.changeEmail = this.changeEmail.bind(this)
14      this.changeUsername = this.changeUsername.bind(this)
15      this.changePassword = this.changePassword.bind(this)
16      this.onSubmit = this.onSubmit.bind(this);
17    }
18
19
20    changeUsername(event){
21      this.setState({
22         username:event.target.value
23      })
24    }
25    changePassword(event){
26      this.setState({
27          password:event.target.value
28      })
29    }
30    changeEmail(event){
31      this.setState({
32          email:event.target.value
33      })
34    }
35    onSubmit(event){
36      event.preventDefault()
37      const registered= {
38          username:this.state.username,
39          email:this.state.email,
```

```
40          password:this.state.password
41      }
42      axios.post ('http://localhost:3001/api/signup', registered)
43        .then(response => {
44          console.log(response.data)
45        })
46        .then(response => {
47          alert('Register successful')
48          window.location.href = '/'
49
50        })
51
52
53
54    }
55
56    render() {
57      return(
58        <div>
59          <div className='container'>
60            <div className='form-div'>
61              <form>
62                <input type='text'
63                placeholder='Username'
64                onChange={this.changeUsername}
65                value={this.state.username}
66                className='form-control form-group'
67                />
68
69                <input type='text'
70                placeholder='E-mail'
71                onChange={this.changeEmail}
72                value={this.state.email}
73                className='form-control form-group'
74                />
75
76                <input type='password'
77                placeholder='Password'
78                onChange={this.changePassword}
79                value={this.state.password}
80                className='form-control form-group'
81                />
82               <button onClick={this.onSubmit}>submit</button>
83            </form>
84          </div>
85        </div>
86      </div>
87      )
88    }
89  }
```

```
90
91 export default Signup;
```

technologies.js:

```
1  import React, { Component } from "react";
2  import Slide from "react-reveal";
3
4  class Technologies extends Component {
5    render() {
6      if (!this.props.data) return null;
7
8      const frameworks = this.props.data.frameworks.map(function (frameworks) {
9        return (
10         <div key={frameworks.name}>
11           <h3>{frameworks.name}</h3>
12
13           <p>{frameworks.description}</p>
14         </div>
15       );
16     });
17
18     const other = this.props.data.other.map(function (other) {
19       return (
20         <div key={other.name}>
21           <h3>{other.name}</h3>
22
23           <p>{other.description}</p>
24         </div>
25       );
26     });
27
28
29
30     return (
31       <section id="technologies">
32         <Slide left duration={1300}>
33
34           <div className="row frameworks">
35             <div className="three columns header-col">
36               <h2>
37                 <span>Frameworks</span>
38               </h2>
39             </div>
40
41             <div className="nine columns main-col">
42               <div className="row item">
43                 <div className="twelve columns">{frameworks}</div>
```

```
              </div>
            </div>
          </div>
        </Slide>

        <Slide left duration={1300}>
          <div className="row other">
            <div className="three columns header-col">
              <h2>
                <span>Others</span>
              </h2>
            </div>

            <div className="nine columns ↙
               ↳ main-col">{other}</div>
          </div>
        </Slide>


      </section>
    );
  }
}

export default Technologies;
```

# L Training and Testing Data

To access the data used to train and test the model here is a shared OneDrive folder with the data in:
https://cranfield-my.sharepoint.com
By signing in with your Cranfield University Email you should be able to access the data.
There are 5 directories within this repository:

- CFD: Contains the Mesh and data files necessary to run the Simulations.
- final_images: The full set of image files used to train the model and evaluate the results.
- final_vortex_locations: The manually labelled set of vortex locations used to train the models.
- vortex_cores: The manually labelled vortex cores for a subset of the image files.
- SIFT ML: A file containing all the data used for the training and figures generated for the report. The .txt files contain the SIFT features used in the training and testing of the model indicated in the name.

# M Timesheet Documents


**Group details**

VvortexX - Group 4

**Student details**

Daniel Smythe

**Week**

1

**Timesheet period**

Period start 04/03/2022

Period end 10/03/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 08/03/2022 | 14:30:00 | 17:30:00 | Began Research into the Yolo Algorithm | 3.00 |
| | 19:00:00 | 21:00:00 | Continued the Yolo Algorithm Research | 2.00 |
| 09/03/2022 | 13:00:00 | 15:00:00 | Looked into ML implementation given in lectures | 2.00 |
| 10/03/2022 | 19:00:00 | 21:00:00 | Set up use of GPU for running Pytorch | 2.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

VvortexX - Group 4

**Student details**

Daniel Smythe

**Week**

2

**Timesheet period**

Period start 11/03/2022

Period end 17/03/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 12/03/2022 | 14:30:00 | 17:30:00 | Began research into commonly used features for feature matching and object tracking | 3.00 |
| 12/03/2022 | 18:30:00 | 20:00:00 | Began testing with the example image files in OpenCV | 1.50 |
| 13/03/2022 | 13:00:00 | 16:00:00 | Read through Szeliski's section on CNNs | 3.00 |
| 15/03/2022 | 12:00:00 | 18:30:00 | Began reading through corrosion paper and reading around the references it provided | 6.50 |
| 16/03/2022 | 13:00:00 | 15:30:00 | Made a simple c++ program to label the vortex image regions | 2.50 |
| 16/03/2022 | 16:00:00 | 17:00:00 | Continued research into CNNs and different architecutres | 1.00 |
| 16/03/2022 | 19:00:00 | 20:30:00 | Met up in the AI group to discuss research and create a plan for the project | 1.50 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

VvortexX - Group 4

**Student details**

Daniel Smythe

**Week**

3

**Timesheet period**

Period start 18/03/2022

Period end 24/03/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 19/03/2022 | 14:00:00 | 18:00:00 | Augmented the Labelling program to additionally preprocess initial image | 4.00 |
| 20/03/2022 | 12:00:00 | 14:00:00 | Began looking at how the images were split up in the Corrosion paper and for ZFNet | 2.00 |
| 20/03/2022 | 18:00:00 | 20:00:00 | Implemented the ZFNet's random cropping in the labelling program | 2.00 |
| 22/03/2022 | 12:00:00 | 17:00:00 | Ported the C++ labelling program's image processing to a python program and began | 5.00 |
| | | | experimentation of contour detection and Fourier Space analysis on vortex images | |
| 22/03/2022 | 19:00:00 | 20:00:00 | Finished the image processing testing of Fourier Space analysis and contour detection | 1.00 |
| 23/03/2022 | 11:00:00 | 12:00:00 | Reread up on SIFT features and the Sobel filters | 1.00 |
| 23/03/2022 | 14:00:00 | 18:00:00 | Began implementing SIFT Feature constructor using the gradients found by Sobel Filters | 4.00 |
| 23/03/2022 | 19:30:00 | 20:30:00 | Tested SIFT feature appearance over vortex regions and non-vortex regions. | 1.00 |
| 24/03/2022 | 17:00:00 | 18:00:00 | Fixed a few bugs in the SIFT feature program | 1.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

VvortexX - Group 4

**Student details**

Daniel Smythe

**Week**

4

**Timesheet period**

Period start 25/03/2022

Period end 31/03/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 25/03/2022 | 09:00:00 | 13:00:00 | Improved upon the SIFT feature generation, including fixing some bug issues, | 4.00 |
| | | | normalising the SIFT results and using the rotational invariance | 0.00 |
| 29/03/2022 | 16:00:00 | 18:30:00 | Collected complete image dataset and produced a program to generate a dataset | 2.50 |
| | | | of pre-processed images for experimentation with using YOLO | 0.00 |
| 29/03/2022 | 19:30:00 | 21:00:00 | Had a call within AI team to discuss current progress and future steps | 1.50 |
| 30/03/2022 | 13:00:00 | 14:00:00 | Generated a quick framework for using Pytorch's neural networks with SIFT feautres | 1.00 |
| | 16:00:00 | 19:30:00 | Cleaned up the labelling python program to allow for mistakes and to output the results | 3.50 |
| | 21:00:00 | 21:30:00 | Fixed a bug in how the results were being stored | 0.50 |
| 31/03/2022 | 11:00:00 | 12:00:00 | Augmented the Labelling program to also produce multiple sub-images at | 1.00 |
| | | | labelled regions | 0.00 |
| | 12:30:00 | 14:30:00 | Fixed a bug in the sub-image generation to produce the correct crops at image borders | 2.00 |
| | | | when randomly selecting a point in the surrounding region | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

VvortexX - Group 4

**Student details**

Daniel Smythe

**Week**

5

**Timesheet period**

Period start 01/04/2022

Period end 07/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 01/04/2022 | 13:00:00 | 14:30:00 | Labelled 700 image files for vortices | 1.50 |
| 02/04/2022 | 12:00:00 | 16:00:00 | Created a program to generate vortex and non-vortex images for classification, then ran it | 4.00 |
| 02/04/2022 | 16:30:00 | 18:00:00 | Reformatted a program to generate ML algoritm based on Sift Features | 1.50 |
| 02/04/2022 | 18:00:00 | 19:00:00 | Began training of ML model | 1.00 |
| 02/04/2022 | 20:00:00 | 22:00:00 | Applied model to image files with, however encountered multiple issues with the code | 2.00 |
| 03/04/2022 | 11:00:00 | 17:00:00 | Assumed issue was with the model's training, so retested the model with different parameters | 6.00 |
| 03/04/2022 | 18:30:00 | 21:00:00 | Re-examined code, found the bugs and managed to get the SIFT feature model to work | 2.50 |
| 04/04/2022 | 15:00:00 | 16:00:00 | Made code to reformat the vortex regions for Yolo algorithm | 1.00 |
| 05/04/2022 | 14:00:00 | 16:00:00 | Remade the program for sub-image generation to also produce the corresponding vortex regions | 2.00 |
| 05/04/2022 | 18:00:00 | 21:00:00 | Began testing the SIFT feature model with differently sized SIFT features and neural networks | 3.00 |
| 06/04/2022 | 12:00:00 | 21:00:00 | Spent the majority of the day waiting on the ML to train and test with different starting parameters. | 9.00 |
| | | | Parameters varied included the SIFT feature sizes, image sizes and neural network node counts. | 0.00 |
| | | | Found that the algorithm works best when the feature sizes roughly match the size of the vortices | 0.00 |
| | | | and when we just use four boxes to describe an image, one for each corner. | 0.00 |


**Group details**

VvortexX - Group 4

**Student details**

Daniel Smythe

**Week**

6

**Timesheet period**

Period start 08/04/2022

Period end 14/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 10/04/2022 | 18:00:00 | 22:00:00 | Began Experimenting with a program to locate vortex cores using gradient directional information from sobel | 4.00 |
| 11/04/2022 | 09:30:00 | 10:30:00 | Finished off core located based on a random voting method before supervised meeting | 1.00 |
| 11/04/2022 | 20:00:00 | 21:00:00 | Remade program to produce the potential vortex regions from the ML model | 1.00 |
| 12/04/2022 | 16:00:00 | 18:00:00 | Generated results of the SIFT region detection from the image set | 2.00 |
| 12/04/2022 | 20:00:00 | 22:00:00 | Augmented the vortex core location to additionally fit the vote results to a gaussian distribution to find the cores | 2.00 |
| 13/04/2022 | 12:00:00 | 14:00:00 | Generated and ran program to manually generate the vortex core locations for testing | 2.00 |
| 13/04/2022 | 15:00:00 | 21:00:00 | Began testing of different parameters to find the optimal value for the vortex core finding | 6.00 |
| 14/04/2022 | 11:00:00 | 12:30:00 | Created a non-random method of finding vortex core locations as an alternative to the random method | 1.50 |
| 14/04/2022 | 12:30:00 | 14:30:00 | Tested the non-random method with different parameters to test against the random methods | 2.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

VvortexX - Group 4

**Student details**

Daniel Smythe

**Week**

7

**Timesheet period**

Period start 15/04/2022

Period end 21/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 16/04/2022 | 13:00:00 | 16:30:00 | Met up and Incorperated the core finding into the post-processing of the YOLO algorithm | 3.50 |
| 16/04/2022 | 20:00:00 | 21:30:00 | Labelled vortex locations in 300 images | 1.50 |
| 17/04/2022 | 13:00:00 | 16:30:00 | Began labelling some of the 1000 image files | 3.50 |
| 17/04/2022 | 19:30:00 | 22:00:00 | Finished labelling the image files that I had at the time | 2.50 |
| 18/04/2022 | 12:00:00 | 13:30:00 | Labelled 400 images that were sent the night before | 1.50 |
| 18/04/2022 | 15:00:00 | 17:00:00 | Worked with the team to plan and produce the initial poster design | 2.00 |
| 18/04/2022 | 20:00:00 | 22:00:00 | Rewrote the core finidng application to incorperate numpy processing to speed it up | 2.00 |
| 18/04/2022 | 22:00:00 | 23:30:00 | Spent some time generated suitable images and animations to incorperate into the presentation | 1.50 |
| 19/04/2022 | 11:00:00 | 13:30:00 | Reworked some programs to generate some suitable figures for the presentation | 2.50 |
| 19/04/2022 | 14:00:00 | 18:00:00 | Met up and worked with the team to finalise our poster design and presentation | 4.00 |
| 19/04/2022 | 20:00:00 | 23:00:00 | Spent some time reworking my portion of the presentation using the generated images | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

VvortexX

**Student details**

Wang David

**Week**

Enter week number 1

**Timesheet period**

Period start 07/03/2022

Period end 13/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 07/03/2022 | 00:14:00 | 00:20:00 | Research on specific libraries,frameworks for UI and backend-code | 6.00 |
| 08/03/2022 | 00:14:00 | 00:20:00 | Link between python back-end code and UI Javascript | 6.00 |
| 09/03/2022 | 00:14:00 | 00:20:00 | Looking at how to upload pictures on UI and store it in DJANGO database | 6.00 |
| 10/03/2022 | 00:14:00 | 00:20:00 | Start implementing UI | 6.00 |
| 11/03/2022 | 00:14:00 | 00:20:00 | Deployment first mockup of web application | 6.00 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

VvortexX

**Student details**

Wang David

**Week**

Enter week number 2

**Timesheet period**

Period start 14/03/2022

Period end 17/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 14/03/2022 | 00:14:00 | 00:18:00 | Research on code architecture | 4.00 |
| 15/03/2022 | 00:14:00 | 00:18:00 | Research on simple ML algorithm | 4.00 |
| 16/03/2022 | 00:14:00 | 00:18:00 | Research on Django models adapted to ML | 4.00 |
| 17/03/2022 | 00:14:00 | 00:18:00 | Start a mockup implementation of a ML model with django | 4.00 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |


**Group details**

VvortexX

**Student details**

Wang David

**Week**

Enter week number 3

**Timesheet period**

Period start 21/03/2022

Period end 24/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 21/03/2022 | 00:14:00 | 00:18:00 | Progress in implementing ML model with django | 4.00 |
| 22/03/2022 | 00:14:00 | 00:18:00 | Progress in implementing ML model with django | 4.00 |
| 23/03/2022 | 00:14:00 | 00:18:00 | solving issue : how to handles queries and routes Django server to machine learning algorithms | 4.00 |
| 24/03/2022 | 00:14:00 | 00:18:00 | Discuss with AI on what format/file they gave us to implement | 4.00 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |


**Group details**

VvortexX

**Student details**

Wang David

**Week**

Enter week number 4

**Timesheet period**

Period start 28/03/2022

Period end 31/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 28/03/2022 | 00:14:00 | 00:18:00 | Progress in UI : Routes different component for different feature | 4.00 |
| 29/03/2022 | 00:14:00 | 00:18:00 | upgrade in UI : Button and others class | 4.00 |
| 30/03/2022 | 00:14:00 | 00:18:00 | Work with AI : resolving environment setup problem | 4.00 |
| 31/03/2022 | 00:14:00 | 00:18:00 | reasearch flask vs django | 4.00 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |


**Group details**

VvortexX

**Student details**

Wang David

**Week**

Enter week number 5

**Timesheet period**

Period start 04/04/2022

Period end 07/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 04/04/2022 | 00:14:00 | 00:18:00 | Finish implementing ML from AI to first web application mockup | 4.00 |
| 05/04/2022 | 00:14:00 | 00:18:00 | UI improvment at : https://whispering-harbor-38933.herokuapp.com | 4.00 |
| 06/04/2022 | 00:14:00 | 00:18:00 | UI improvment at : https://whispering-harbor-38933.herokuapp.com | 4.00 |
| 07/04/2022 | 00:14:00 | 00:18:00 | Looking how to develop user authentification feature | 4.00 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |


**Group details**

VvortexX

**Student details**

Wang David

**Week**

Enter week number 6

**Timesheet period**

Period start 11/04/2022

Period end 14/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 11/04/2022 | 00:14:00 | 00:18:00 | Connect application to Database create scheme / model | 4.00 |
| 12/04/2022 | 00:14:00 | 00:18:00 | fetch users data | 4.00 |
| 13/04/2022 | 00:14:00 | 00:18:00 | print information on website | 4.00 |
| 14/04/2022 | 00:14:00 | 00:18:00 | work on authentification feature | 4.00 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |


**Group details**

VvortexX

**Student details**

Wang David

**Week**

Enter week number 7

**Timesheet period**

Period start 18/04/2022

Period end 21/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 18/04/2022 | 00:14:00 | 00:18:00 | Link second version of the detection feature | 4.00 |
| 19/04/2022 | 00:14:00 | 00:18:00 | work with AI for tracking feature | 4.00 |
| 20/04/2022 | 00:14:00 | 00:18:00 | UI design | 4.00 |
| 21/04/2022 | 00:14:00 | 00:18:00 | Create model of feature utilisation and store it in database | 4.00 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |


**Group details**

Group 4 | VvortexX

**Student details**

Imad Ul Islam

**Week**

1 1

**Timesheet period**

Period start 3/4/2021

Period end 3/11/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 3/1/2021 | 09:30:00 | 13:30:00 | Presentation | 4.00 |

| | | | | |
|---|---|---|---|---|
| 3/4/2021 | 13:00:00 | 21:00:00 | Group meeting + Research | 8.00 |
| 3/5/2021 | 16:00:00 | 21:00:00 | Research on Fluid Dynamics(Turbulence) | 5.00 |
| 3/6/2021 | 17:00:00 | 20:00:00 | Research on Turbulence(Vortices) | 3.00 |
| 3/8/2021 | 16:00:00 | 21:00:00 | Research on methodology of project | 5.00 |
| 3/10/2021 | 20:00:00 | 00:00:00 | Group meeting | 4.00 |
| 11/03/2021 | 10:00:00 | 13:00:00 PM | Research | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Imad Ul Islam

**Week**

2

**Timesheet period**

Period start 3/12/2021

Period end 3/17/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 3/13/2021 | 11:30:00 | 13:30:00 | Creating a problem statement | 2.00 |

| | | | | |
|---|---|---|---|---|
| 3/14/2021 | 13:00:00 | 17:00:00 | esearch on methods to solve the probler | 4.00 |
| 3/15/2021 | 16:00:00 | 21:00:00 | Research on Fluid Dynamics | 5.00 |
| 3/16/2021 | 17:00:00 | 20:00:00 | arch on Turbulence(Vortices)+online me | 3.00 |
| 3/17/2021 | 16:00:00 | 20:00:00 | Group meeting and research | 8.00 |
| | | | | 0.00 |
| | | | | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Imad Ul Islam

**Week**

3

**Timesheet period**

Period start 3/19/2022

Period end 3/25/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 3/19/2022 | 11:00:00 | 14:00:00 PM | ‹ing the dependencies of the object dete | 3.00 |
| 3/20/2021 | 13:00:00 | 17:00:00 | fixing the dependencies of the model | 4.00 |
| 3/21/2021 | 2:00:00 PM | 16:00:00 | Group meeting | 3.00 |
| 3/22/2021 | 17:00:00 | 22:00:00 | installing cocopytools and fixing bugs in | 3.00 |
| 3/23/2021 | 16:00:00 | 20:00:00 | reading articles on object detection | 4.00 |
| 3/24/2021 | 13:00:00 | 18:00:00 | group meeting | 5.00 |
| | | | | |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |

| | |
|---|---|
| | 0.00 |
| | 0.00 |
| | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Imad Ul Islam

**Week**

5

**Timesheet period**

Period start 4/1/2021

Period end 4/7/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 4/1/2021 | 10:30:00 | 18:30:00 PM | ıg a dataset and training the model for t | 8.50 |

| | | | | |
|---|---|---|---|---|
| 4/2/2021 | 10:00:00 | 22:00:00 PM | testing the model with different image | 12.00 |
| 4/3/2021 | 12:00:00 | 21:00:00 | testing different image pre processing | 9.00 |
| 4/4/2021 | 9:00:00 AM | 20:00:00 | p Meeting + recap of the project progres | 11.00 |
| 4/5/2021 | 16:00:00 | 21:00:00 | Researching content for the report | 5.00 |
| 4/6/2021 | 11:00:00 | 15:00:00 PM | Fixing bugs in the model | 4.00 |
| 07/04/2021 | 09:00:00 | 20:00:00 PM | p meeting + Comparing the speed of mo | 11.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Imad Ul Islam

**Week**

6

**Timesheet period**

Period start 4/7/2021

Period end 4/14/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 4/8/2021 | 12:30:00 | 18:30:00 PM | Tuning selected models | 6.00 |

| | | | | |
|---|---|---|---|---|
| 4/9/2021 | 11:00:00 | 22:00:00 PM | and testing the model with different pro | 11.00 |
| 4/10/2021 | 13:00:00 | 19:00:00 | building an api | 6.00 |
| 4/11/2021 | 9:00:00 AM | 20:00:00 | p Meeting + recap of the project progres | 11.00 |
| 4/12/2021 | 17:00:00 | 20:00:00 | Researching content for the report | 3.00 |
| 4/13/2021 | 07:00:00 | 14:00:00 PM | Fixing bugs in the model | 8.00 |
| 14/04/2021 | 15:00:00 PM | 20:00:00 PM | Group meeting + bug fixing | 5.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Imad Ul Islam

**Week**

7

**Timesheet period**

Period start 4/15/2021

Period end 4/21/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 4/15/2021 | 13:30:00 | 18:30:00 PM | Adding features to the project | 5.00 |

| | | | | |
|---|---|---|---|---|
| 4/16/2021 | 13:00:00 | 19:00:00 PM | and testing the model with different pro | 11.00 |
| 4/17/2021 | 13:00:00 | 19:00:00 | Group meeting (AI) + combining the code | 6.00 |
| 4/18/2021 | 11:00:00 AM | 15:00:00 PM | Group meeting | 4.00 |
| 4/19/2021 | 17:00:00 | 20:00:00 | Working on poster | 3.00 |
| 4/20/2021 | 06:00:00 | 12:00:00 PM | Fixing bugs and working on presentation | 6.00 |
| 21/04/2021 | 15:00:00 PM | 20:00:00 PM | Group meeting | 5.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name

Group 4 VvortexX

**Student details**

Enter Student name

Itohan Eregie

**Week**

Enter week number 1

**Timesheet period**

Period start ##########

Period end 11/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 07/03/2022 | 09:00:00 | 13:00:00 | Revision of the presentations of the modules presented by the facilitators to better understand the techniques the assignment | 4.00 |
| | 13:00:00 | 17:00:00 | I configured and setup Jira software for the planning and management of the project. I added all the stake holders for the project and created Epics | 4.00 |
| 08/03/2022 | 09:00:00 | 14:00:00 | Study of the architecture(microservices) and how several parts of the application will be connected | 5.00 |
| | 14:00:00 | 16:00:00 | I did some study about APIs and how they can be written and called by several parts of the application. This is necessary for the micro services architecture we are adopting | 2.00 |
| 09/03/2022 | 09:00:00 | 13:00:00 | Continued study of the micro services architecture | 4.00 |
| | 13:00:00 | 16:00:00 | Study of vortices and how they are formed(Research on the project) | 3.00 |
| 10/03/2022 | 09:00:00 | 13:00:00 | I researched on how devops can be applied to the software by configuring build pipelines for automated testing and deployment | 4.00 |
| | 13:00:00 | 15:00:00 | I configured git two repository for the machine learning module and software module and linked them to circle CI build server for automated build , testing and deployment | 2.00 |
| | 15:00:00 | 19:00:00 | We had a group meeting where we further discussed the achitecture and how the APIs can be called from the frontend software module | 4.00 |
| 11/03/2022 | 10:00:00 | 14:00:00 | I carried out a test configuration of an automated deployment by configuring a build pipeline with test automation . This is in preparation for when the first iteration of the project is ready to be tested | 4.00 |
| | 15:00:00 | 19:00:00 | I carreid out a first test and implementation on how to write and call an API in python . I further outlined and flesed out the archetecture of the whole solution | 4.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details** group 4

Enter Group number and n vvortexx

Group 4 VvortexX

**Student details** 5356017

Enter Student name Itohan Eregie

Itohan Eregie

**Week**

Enter week number 2

**Timesheet period**

Period start ##########

Period end 11/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 14/03/2022 | 09:00:00 | 11:00:00 | Self study on project management on Jira to better organize the tasks | 2.00 |
| | 13:00:00 | 17:00:00 | Test containerization of python app | 4.00 |
| 15/03/2022 | 09:00:00 | 12:00:00 | More study on microservices | 3.00 |
| | 14:00:00 | 16:00:00 | I did some study about APIs and how they can be written and called by several parts of the application. This is necessary for the micro services architecture we are adopting | 2.00 |
| 16/03/2022 | 09:00:00 | 13:00:00 | I watched some tutorials on circle CI , the build tool for the continuous integration and deployment | 4.00 |
| | 13:00:00 | 14:00:00 | I reorganized Jira and defined the tasks more clearly | 1.00 |
| 17/03/2022 | 11:00:00 | 13:00:00 | Test configuration on circle CI , creating build pipelines for automated build | 2.00 |
| | | | | |
| | | | | |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name: vvortex Group4

**Student details**

Enter Student name: Itohan Eregie s356017

**Week**

Enter week number: 3

**Timesheet period**

Period start: 18/03/2021

Period end: 24/03/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 18/03/2022 | | | | |
| | 14:00:00 | 16:00:00 | | 2.00 |
| 21/03/2022 | 09:00:00 | 16:00:00 | inued with the buildpipeline on circle CI and Did some more tutotials . I Tried it with the actual front end code but it was f | 7.00 |
| | | | | 0.00 |
| 22/03/2022 | 09:00:00 | 12:00:00 | some progress configuring the buildpipeline to work , it was partly successful. Some configurations are missing from the | 3.00 |
| | 12:00:00 | 00:00:00 | uilding a mockup project to understand the frame work, then applying it to the code we have .This will enable me docum | 12.00 |
| 23/03/2022 | 09:00:00 | 18:00:00 | I started documenting the software requirements | 9.00 |
| | | | | 0.00 |
| 24/03/2022 | 08:00:00 | 13:00:00 | ing on how to properly configure the buidpipeline made a head way . The front end code built successfully . It can be dep | 5.00 |
| | 13:00:00 | 15:00:00 | I attended team meeting | 2.00 |
| | 15:00:00 | 23:00:00 | I continued with documenting the software requirements document | 8.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details** group 4

Enter Group number and n vvortexx

Group 4 VvortexX

**Student details** 5356017

Enter Student name Itohan Eregie

Itohan Eregie

**Week**

Enter week number 4

**Timesheet period**

Period start ##########

Period end 11/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 25/03/2022 | 09:00:00 | 15:00:00 | I continued the documentation of the software requirements | 6.00 |
| | 15:00:00 | 17:00:00 | I continued working on the front end code figuring out how to link the front end with the machine learning module | 2.00 |
| 26/03/2022 | 09:00:00 | 12:00:00 | e we had again and tried to understand it and model it to what we already have . I also suggested to the team that we have a group code peer review from top to bottom . I also led the team in the microse | 3.00 |
| | 14:00:00 | 18:00:00 | was necessary to understand how to call the ML API service. This is also important to properly containerize each module as a micro service. I also researched and found simpler implementation of how to | 4.00 |
| 27/03/2022 | 12:00:00 | 14:00:00 | mple implementation first before improving on it. During the meeting, I explained microservices architecture and API to the team .At the outcome of the meeting , we were able to know the actual output fr | 2.00 |
| | | | | 0.00 |
| 28/03/2022 | 09:00:00 | 17:00:00 | I was able to build the first docker image for the front end . Though the container did not run but it was first attempt , it is available within docker hub | 8.00 |
| | | | | |
| 29/03/2022 | 09:00:00 | 18:00:00 | hooting the docker containerization of the front end application . This container is still not running . I pulleded an empty container from docker hub and tried to copy the front end code into it to test , but it still was not working | |
| | | | | 0.00 |
| 30/03/2022 | 09:00:00 | 14:00:00 | I continued with the software requirements documentation and comoleted the first version . I also adjusted the project plan on confluence (it is a add on tool within Jira for project plan) | 5.00 |
| | | | | 0.00 |
| 31/03/2022 | 09:00:00 | 02:00:00 | I continued to look into how to connect the front end application with the back end api | 17.00 |
| | 15:00:00 | 16:00:00 | I attended group meeting | 1.00 |
| | | | | 0.00 |


**Group details** group 4

Enter Group number and n vvortexx

Group 4 VvortexX

**Student details** 5356017

Enter Student name Itohan Eregie

Itohan Eregie

**Week**

Enter week number 6

**Timesheet period**

Period start ##########

Period end 11/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 01/03/2022 | 09:00:00 | 15:00:00 | ion with lmad on tensor flow. I discussed the flow of the code and how the ML module links with the software front end . I did this to properly architect the deployment of the solution in microservices archi | 6.00 |
| | | | | 0.00 |
| 04/03/2022 | 15:00:00 | 20:00:00 | continuation of software requirements documentation. I finished the first draft of this document . It wil be reviewed by the group on Thursday 7th of April in our meeting | 5.00 |
| | | | | 0.00 |
| 05/03/2022 | 09:00:00 | 14:00:00 | I started documenting the software architecture document | 5.00 |
| | 14:00:00 | 17:00:00 | I made adjustments to the project plan highlihgting the risks we faced during our first sprint. | 3.00 |
| 06/03/2022 | 09:00:00 | 17:00:00 | I started trying to deploy the application in a microservices method. The framework for the first mock up we did was in django. The second and final framework is in flask. | 8.00 |
| | | | | |
| 07/03/2022 | 09:00:00 | 12:00:00 | Continues attempt to deploy the application as microservices on heroku . I also started part of the documentation for the software architecture document | |
| | 13:00:00 | 16:00:00 | Group meeting | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details** group 4

Enter Group number and n vvortexx

Group 4 VvortexX

**Student details** 5356017

Enter Student name Itohan Eregie

Itohan Eregie

**Week**

Enter week number 7

**Timesheet period**

Period start ##########

Period end 11/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 07/04/2022 | 09:00:00 | 15:00:00 | I started the documentation for software design and architecture | 6.00 |
| | | | | 0.00 |
| 08/04/2022 | 15:00:00 | 20:00:00 | I started deploying the application as a micro service my attempting to deploy the ML module as a separate service . I configured heroku to deploy the ML module | 5.00 |
| | | | | 0.00 |
| 11/03/2022 | 09:00:00 | 12:00:00 | I attended the Monday review meeting with the supervisors | 3.00 |
| | 12:00:00 | 03:00:00 | We had our teams meeting after that | |
| 12/03/2022 | 09:00:00 | 17:00:00 | I continued with deploying the ML module as a micro service on heroku | 8.00 |
| | | | | |
| 13/14/2022 | 09:00:00 | 12:00:00 | I continued with deploying the ML module as a micro service on heroku | |
| | 13:00:00 | 15:00:00 | Group meeting | 2.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details** group 4

Enter Group number and n vvortexx

Group 4 VvortexX

**Student details** 5356017

Enter Student name Itohan Eregie

Itohan Eregie

**Week**

Enter week number 8

**Timesheet period**

Period start ##########

Period end 11/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 18/04/2022 | 01:00:00 | 07:00:00 | Group meeting with the teamto design poster and discuss the slides for the presentation | 6.00 |
| 19/04/2022 | 01:00:00 | 07:00:00 | Group meetint with team to work on presentation slides and design the poster | 6.00 |
| 20/04/2022 | 02:00:00 | 07:00:00 | Group meeting with the team to mconclude with the poster design and the presentation slides | 5.00 |
| 21/04/2022 | 03:00:00 | 05:00:00 | Group meeting with the team to discuss the litereture review and finalize the prototype | 2.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | |
| | | | | |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| 31/03/2022 | 09:00:00 | 02:00:00 | I continued to look into how to connect the front end application with the back end api | 17.00 |
| | 15:00:00 | 16:00:00 | I attended group meeting | 1.00 |
| | | | | 0.00 |

# Weekly Timesheet
## CSTE Group Project

**Group details**

Group number and name — Group No.4 (VvortexX)

**Student details**

Student name — Pierre Emmanuel-Désiré CISSE

**Week**

Week number 1

**Timesheet period**

Period start 04/03/2022

Period end 11/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| ######### | 09:30:00 | 12:30:00 | Preliminary Group presentation & Supervisors guidance | 3.00 |
| | 13:30:00 | 16:30:00 | General Discussion & Kick-off meeting | 3.00 |
| ######### | 10:00:00 | 14:30:00 | Personal study on the topics covered during the week (ML, CMV) | 4.50 |
| ######### | 10:30:00 | 15:30:00 | Self-Study (Theory behing CFD work, Possible GUI implementation) | 5.00 |
| ######### | 15:00:00 | 18:30:00 | 1st Regular Meeting | 3.50 |
| ######### | 11:30:00 | 14:30:00 | Litterature review of papers according to our chosen test case | 3.00 |
| ######### | 10:30:00 | 14:30:00 | Study APIs implementation + Possible architectures for our final prototype | 4.00 |
| ######### | 15:00:00 | 18:30:00 | 2nd Regular Meeting | 3.50 |
| ######### | 12:30:00 | 18:30:00 | Study Docker implementation + Formating the documents to be used during Regular Meetings | 6.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |

# Weekly Timesheet
## CSTE Group Project

**Group details**

Group number and name — Group No.4 (VvortexX)

**Student details**

Student name — **Pierre Emmanuel-Désiré CISSE**

**Week**

Week number 2

**Timesheet period**

Period start 12/03/2022

Period end 17/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| ######### | 09:30:00 | 12:30:00 | As leader and SE: research on some team performance study tools to be automate via JIRA | 3.00 |
| | 13:30:00 | 16:30:00 | More research on JIRA for the generation of detailed reports on the project plan structure | 3.00 |
| ######### | 09:30:00 | 11:30:00 | Discussion with supervisors on team progress + Questions (Weekly meeting) | 2.00 |
| | 12:00:00 | 13:30:00 | Regular Meeting (In person) | 1.50 |
| | 15:00:00 | 18:30:00 | Research difference between CFD solvers | 3.50 |
| ######### | 11:30:00 | 14:30:00 | Litterature review of papers according to possible architectures | 3.00 |
| ######### | 14:30:00 | 16:30:00 | Meeting within SE sub-group to discussion documentation + plan | 2.00 |
| | 18:00:00 | 21:30:00 | Study APIs implementation + Study ML theory (CNN) | 3.50 |
| ######### | 11:00:00 | 13:00:00 | Regular Meeting (In person) | 2.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group number and name: Group No.4 (VvortexX)

**Student details**

Student name: **Pierre Emmanuel-Désiré CISSE**

**Week**

Week number: 3

**Timesheet period**

Period start: 18/03/2022

Period end: 24/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| ######### | 09:30:00 | 12:30:00 | As CFD: research on running ANSYS/ Fluent on Crescent and Help doing so | 3.00 |
| | 13:30:00 | 16:30:00 | More research on CFD solvers + literature review on CFD theory | 3.00 |
| ######### | 09:30:00 | 11:30:00 | Study ML model implementation on microservices | 2.00 |
| | 12:00:00 | 13:30:00 | Start writting CFD Theory | 1.50 |
| ######### | 09:30:00 | 11:00:00 | Discussion with supervisors on team progress + Questions (Weekly meeting) | 1.50 |
| | 11:00:00 | 12:00:00 | Regular Meeting (In person) | 1.00 |
| | 14:00:00 | 16:30:00 | Regular Meeting (In person) | 2.50 |
| ######### | 18:00:00 | 21:30:00 | Study APIs implementation + Doing tutorial of ML implementation (CNN) | 3.50 |
| ######### | 14:00:00 | 15:30:00 | CFD Meeting (Online : teams) | 1.50 |
| ######### | 13:30:00 | 16:30:00 | Regular Meeting (In person) | 3.00 |
| | 18:00:00 | 21:30:00 | Check the progress of the different sub-teams and give organizational directives for the group + Work on the n | 3.50 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |

# Weekly Timesheet
## CSTE Group Project

**Group details**

Group number and name — Group No.4 (VvortexX)

**Student details**

Student name — **Pierre Emmanuel-Désiré CISSE**

**Week**

Week number 4

**Timesheet period**

Period start 25/03/2022

Period end 31/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| ######### | 14:30:00 | 17:30:00 | As SE: research on connecting the AI model with the SE mockup (Trial with our own ML-test-model) | 3.00 |
| | 20:30:00 | 22:30:00 | More research on data management of the prototype | 2.00 |
| ######### | 09:30:00 | 11:30:00 | Study ML model implementation on microservices | 2.00 |
| | 12:00:00 | 13:30:00 | Writting CFD Theory | 1.50 |
| ######### | 09:30:00 | 11:00:00 | Discussion with supervisors on team progress + Questions (Weekly meeting) | 1.50 |
| | 12:00:00 | 13:30:00 | Regular Meeting (In person) | 1.50 |
| | 14:00:00 | 16:30:00 | Writting CFD Theory | 2.50 |
| ######### | 18:00:00 | 21:30:00 | Study CFD solvers | 3.50 |
| ######### | 14:00:00 | 15:30:00 | CFD documents improvement | 1.50 |
| ######### | 15:00:00 | 16:30:00 | Regular Meeting (In person) | 1.50 |
| | 18:00:00 | 21:30:00 | Check the progress of the different sub-teams and give organizational directives for the group + Work on the ma | 3.50 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |

# Weekly Timesheet
## CSTE Group Project

**Group details**

Group number and name — Group No.4 (VvortexX)

**Student details**

Student name — Pierre Emmanuel-Désiré CISSE

**Week**

Week number 5

**Timesheet period**

Period start 01/04/2022

Period end 07/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| ######### | 14:30:00 | 17:00:00 | As CFD: research on Fluent and the meaning of some features (some solvers) | 2.50 |
| | 20:00:00 | 22:30:00 | More research on Fluent and run simulations from VP | 2.50 |
| ######### | 09:30:00 | 11:30:00 | Study ML model implementation on microservices | 2.00 |
| | 12:00:00 | 13:30:00 | Writting CFD Theory | 1.50 |
| ######### | 09:30:00 | 11:00:00 | Discussion with supervisors on team progress + Questions (Weekly meeting) | 1.50 |
| | 11:00:00 | 13:00:00 | Regular Meeting (In person) | 2.00 |
| | 14:00:00 | 16:30:00 | Writting CFD Theory | 2.50 |
| ######### | 18:00:00 | 21:30:00 | Study CFD solvers and try to run new simulations given by VP | 3.50 |
| ######### | 14:00:00 | 15:30:00 | CFD documents improvement | 1.50 |
| ######### | 13:00:00 | 16:00:00 | Regular Meeting (In person) | 3.00 |
| | 18:00:00 | 21:30:00 | Check the progress of the different sub-teams and give organizational directives for the group + Work on the ma | 3.50 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |

# Weekly Timesheet
## CSTE Group Project

**Group details**

Group number and name — Group No.4 (VvortexX)

**Student details**

Student name — **Pierre Emmanuel-Désiré CISSE**

**Week**

Week number 6

**Timesheet period**

Period start 08/04/2022

Period end 14/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| ######### | 14:30:00 | 17:00:00 | Research on CFD | 2.50 |
| | 20:00:00 | 22:30:00 | Run simulations from VP | 2.50 |
| ######### | 09:30:00 | 11:30:00 | Study ML model implementation on microservices | 2.00 |
| | 12:00:00 | 13:30:00 | Writting CFD Theory | 1.50 |
| ######### | 09:30:00 | 12:00:00 | Discussion with supervisors on team progress + Questions (Weekly meeting) | 2.50 |
| | 12:00:00 | 14:00:00 | Regular Meeting (In person) | 2.00 |
| | 14:00:00 | 16:30:00 | Writting Quality Strategy | 2.50 |
| ######### | 18:00:00 | 21:30:00 | Quality document improvement | 3.50 |
| ######### | 14:00:00 | 15:30:00 | CFD documents improvement | 1.50 |
| ######### | 16:00:00 | 18:00:00 | Regular Meeting (In person) | 2.00 |
| | 18:30:00 | 23:50:00 | Check the progress of the different sub-teams and give organizational directives for the group + Work on the ma | 5.33 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |

# Weekly Timesheet
## CSTE Group Project

**Group details**

Group number and name: Group No.4 (VvortexX)

**Student details**

Student name: Pierre Emmanuel-Désiré CISSE

**Week**

Week number: 7

**Timesheet period**

Period start: 14/04/2022

Period end: 21/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| ######### | 14:30:00 | 17:00:00 | Research on Market Analysis | 2.50 |
| | 20:00:00 | 22:30:00 | Research on Literature Review, Study of Articles | 2.50 |
| ######### | 09:30:00 | 11:30:00 | Quality Document evolvement | 2.00 |
| | 12:00:00 | 13:30:00 | Risk Analysis Document | 1.50 |
| ######### | 09:30:00 | 12:00:00 | Discussion with supervisors on team progress + Questions (Weekly meeting) | 2.50 |
| | 14:20:00 | 19:00:00 | Regular Meeting (In person) | 4.67 |
| | 14:00:00 | 16:30:00 | Writting Quality Strategy | 2.50 |
| ######### | 14:50:00 | 19:00:00 | Regular Meeting (In person) | 4.17 |
| ######### | 14:00:00 | 15:30:00 | CFD documents improvement | 1.50 |
| ######### | 15:00:00 | 17:00:00 | Regular Meeting (In person) | 2.00 |
| | 18:30:00 | 23:50:00 | Check the progress of the different sub-teams and give organizational directives for the group + Work on the ma | 5.33 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name

VORTEX (Group No.4)

**TAEYONG KIM**

**Student details**

Enter Student name

**Week**

Enter week number 1

**Timesheet period**

Period start 04/03/2022

Period end 11/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 04/03/2022 | 09:30:00 | 12:30:00 | Preliminary Group presentation | 3.00 |
| 04/03/2022 | 12:30:00 | 15:30:00 | General Discussion | 3.00 |
| 05/03/2022 | 10:30:00 | 12:30:00 | Self-Study (Vortex, Fluid Analysis) | 2.00 |
| 06/03/2022 | 10:30:00 | 12:30:00 | Self-Study (Convolutional Neural Network) | 2.00 |
| 07/03/2022 | 15:00:00 | 18:00:00 | 1st Regular Meeting | 3.00 |
| 08/03/2022 | 10:30:00 | 14:30:00 | Self-Study (Vortex Papers, Data Source, CNN) | 4.00 |
| 09/03/2022 | 10:30:00 | 14:30:00 | Self-Study (Testing CNN Simple Tutorial) | 4.00 |
| 10/03/2022 | 15:00:00 | 18:00:00 | 2nd Regular Meeting | 3.00 |
| 11/03/2022 | 10:30:00 | 14:30:00 | Self-Study (Testing Matlab Tutorial) | 4.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name

VORTEX (Group No.4)

**TAEYONG KIM**

**Student details**

Enter Student name

**Week**

Enter week number 2

**Timesheet period**

Period start 12/03/2022

Period end 17/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 12/03/2022 | 10:30:00 | 12:30:00 | Self-Study (Vortex, Fluid Analysis) | 2.00 |
| 13/03/2022 | 19:30:00 | 22:30:00 | Self-Study (Image Processing) | 3.00 |
| 14/03/2022 | 09:30:00 | 11:30:00 | Compulsory Regular Meeting | 2.00 |
| 14/03/2022 | 11:30:00 | 13:30:00 | 3rd Regular Meeting | 2.00 |
| 14/03/2022 | 15:30:00 | 18:30:00 | Self-Study (Reviewing Paper) | 3.00 |
| 15/03/2022 | 10:30:00 | 16:30:00 | Self-Study (Coding : Image Processing) | 6.00 |
| 16/03/2022 | 15:00:00 | 18:00:00 | Self-Study (Generating Dataset) | 3.00 |
| 16/03/2022 | 19:00:00 | 21:00:00 | 1st Subgroup Meeting | 2.00 |
| 16/03/2022 | 21:00:00 | 00:00:00 | Self-Study (Paraview : Image Processing) | 3.00 |
| 17/03/2022 | 11:00:00 | 13:00:00 | 4th Regular Meeting | 2.00 |
| 17/03/2022 | 15:00:00 | 18:00:00 | Self-Study (CNN) | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name

VORTEX (Group No.4)

**TAEYONG KIM**

**Student details**

Enter Student name

**Week**

Enter week number 3

**Timesheet period**

Period start 18/03/2022

Period end 24/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 18/03/2022 | 12:30:00 | 18:30:00 | Self-Study (Coding Classification Model, Reviewing Document for Generating | 6.00 |
| 19/03/2022 | 18:30:00 | 22:30:00 | Self-Study (Generating Dataset) | 4.00 |
| 21/03/2022 | 09:30:00 | 12:30:00 | Compulsory Regular Meeting | 3.00 |
| 21/03/2022 | 13:00:00 | 16:00:00 | 5th Regular Meeting | 3.00 |
| 14/03/2022 | 14:30:00 | 17:30:00 | Self-Study (Reviewing Q-criterion) | 3.00 |
| 22/03/2022 | 10:30:00 | 16:30:00 | Self-Study (Testing Q-criterion example) | 6.00 |
| 23/03/2022 | 10:00:00 | 16:00:00 | Self-Study (Matlab related to handling NetCDF format) | 6.00 |
| 23/03/2022 | 19:00:00 | 20:00:00 | technical review meeting (online) | 1.00 |
| 24/03/2022 | 10:00:00 | 12:00:00 | Self-Study (ResNet Model) | 2.00 |
| 24/03/2022 | 13:00:00 | 16:00:00 | 4th Regular Meeting | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name

VORTEX (Group No.4)

**TAEYONG KIM**

**Student details**

Enter Student name

**Week**

Enter week number 3

**Timesheet period**

Period start 25/03/2022

Period end 31/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 25/03/2022 | 12:30:00 | 18:30:00 | Self-Study (Compare Model(YOLO, others) | 6.00 |
| 26/03/2022 | 18:30:00 | 22:30:00 | Self-Study (Generating Flask Environment by using Flask) | 4.00 |
| 28/03/2022 | 09:30:00 | 12:30:00 | Compulsory Regular Meeting | 3.00 |
| 28/03/2022 | 13:00:00 | 16:00:00 | 5th Regular Meeting | 3.00 |
| 29/03/2022 | 10:30:00 | 13:30:00 | Self-Study (Review 4th CFD Result & Generating Dataset) | 3.00 |
| 29/03/2022 | 14:00:00 | 16:00:00 | Self-Study (API Condition) | 2.00 |
| 29/03/2022 | 19:00:00 | 20:00:00 | technical review meeting (online) | 1.00 |
| 30/03/2022 | 10:00:00 | 18:00:00 | Self-Study (Improve YOLO Model) | 8.00 |
| 29/03/2022 | 12:00:00 | 14:00:00 | technical review meeting (online) | 2.00 |
| 31/03/2022 | 15:00:00 | 17:00:00 | 4th Regular Meeting | 2.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name

VORTEX (Group No.4)

**TAEYONG KIM**

**Student details**

Enter Student name

**Week**

Enter week number 5

**Timesheet period**

Period start 01/04/2022

Period end 07/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 01/04/2022 | 13:30:00 | 18:30:00 | Self-Study (YOLO Model theory) | 5.00 |
| 02/04/2022 | 11:00:00 | 14:00:00 | Self-Study (Yolo v4 darknet model test) | 3.00 |
| 03/04/2022 | 18:30:00 | 21:30:00 | Self-Study (Yolo Dataset) | 3.00 |
| 04/04/2022 | 09:30:00 | 12:30:00 | Compulsory Regular Meeting | 3.00 |
| 04/04/2022 | 13:00:00 | 16:00:00 | 7th Regular Meeting | 3.00 |
| 04/04/2022 | 19:30:00 | 21:30:00 | Self-Study (Model trainig based on New dataset) | 2.00 |
| 05/04/2022 | 10:00:00 | 16:00:00 | Self-Study (Dataset error correction and Model | 6.00 |
| 05/04/2022 | 19:00:00 | 20:00:00 | Self-Study (Test History Document Summarization) | 1.00 |
| 06/04/2022 | 10:00:00 | 11:00:00 | Technical review meeting (online) | 1.00 |
| 06/04/2022 | 12:00:00 | 20:00:00 | Self-Study (Model trainig based on New dataset_2) | 8.00 |
| 07/04/2022 | 15:00:00 | 17:00:00 | 8th Regular Meeting | 2.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name

VORTEX (Group No.4)

**TAEYONG KIM**

**Student details**

Enter Student name

**Week**

Enter week number 6

**Timesheet period**

Period start 08/04/2022

Period end 14/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 08/04/2022 | 10:30:00 | 13:30:00 | Self-Study (Documentation Work) | 3.00 |
| 08/04/2022 | 16:00:00 | 19:00:00 | Self-Study (SITF Model to generate coordinates) | 3.00 |
| 10/04/2022 | 18:30:00 | 21:30:00 | Self-Study (Model training based on spinning model) | 3.00 |
| 11/04/2022 | 09:30:00 | 11:30:00 | Compulsory Regular Meeting | 2.00 |
| 11/04/2022 | 13:00:00 | 16:00:00 | 9th Regular Meeting | 3.00 |
| 12/04/2022 | 10:00:00 | 14:00:00 | Self-Study (Model trainig based on New(spinning) dataset) | 4.00 |
| 12/04/2022 | 18:00:00 | 21:00:00 | Self-Study (Presentation and Poster Material) | 3.00 |
| 13/04/2022 | 10:00:00 | 11:00:00 | Self-Study (Test History Document Summarization) | 1.00 |
| 13/04/2022 | 12:00:00 | 17:00:00 | Self-Study (Model trainig based on New(spinning 4k) dataset) | 5.00 |
| 14/04/2022 | 10:00:00 | 13:00:00 | Self-Study (Build comparative model (Faster R-CNN) | 3.00 |
| 14/04/2022 | 15:00:00 | 18:00:00 | 10th Regular Meeting | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Enter Group number and name

VORTEX (Group No.4)

**TAEYONG KIM**

**Student details**

Enter Student name

**Week**

Enter week number 7

**Timesheet period**

Period start 15/04/2022

Period end 21/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 15/04/2022 | 10:30:00 | 13:30:00 | Self-Study (Documentation Work) | 3.00 |
| 15/04/2022 | 16:00:00 | 19:00:00 | Self-Study (Test Model based on different image sizes) | 3.00 |
| 16/04/2022 | 10:30:00 | 12:30:00 | Self-Study (Documentation Work) | 2.00 |
| 16/04/2022 | 14:30:00 | 17:30:00 | Technical Sub Group Meeting | 3.00 |
| 17/04/2022 | 10:30:00 | 18:30:00 | Self-Study (Model training based on new dataset & Prepare presentation) | 8.00 |
| 18/04/2022 | 09:30:00 | 11:30:00 | Self-Study (Model training based on new dataset) | 2.00 |
| 18/04/2022 | 13:00:00 | 18:00:00 | 10th Regular Meeting | 5.00 |
| 19/04/2022 | 10:00:00 | 14:00:00 | Self-Study (Poster material Task) | 4.00 |
| 19/04/2022 | 14:00:00 | 18:00:00 | 11th Regular Meeting | 4.00 |
| 19/04/2022 | 19:00:00 | 22:00:00 | Self-Study (Presentation material Task) | 3.00 |
| 20/04/2022 | 12:00:00 | 17:00:00 | Self-Study (Documentation Work) | 5.00 |
| 21/04/2022 | 10:00:00 | 13:00:00 | Self-Study (Review Poster and Presentation Material) | 3.00 |
| 21/04/2022 | 15:00:00 | 18:00:00 | 12th Regular Meeting | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Vetrichelvan Pugazendi

**Week**

1 1

**Timesheet period**

Period start 04/03/2021

Period end 11/03/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 01/03/2021 | 09:30:00 | 13:30:00 | Presentation | 4.00 |
| 04/03/2021 | 13:00:00 | 00:00:00 | Model preparation, Ansys | 11.00 |
| 05/03/2021 | 16:00:00 | 06:00:00 | Research | 14.00 |
| 06/03/2021 | 17:00:00 | 06:00:00 | Research | 13.00 |
| 08/03/2021 | 16:00:00 | 05:00:00 | Research | 13.00 |
| 10/03/2021 | 20:00:00 | 00:00:00 | Model preparation, Ansys | 4.00 |
| 11/03/2021 | 10:00:00 | 12:00:00 | Model preparation, Ansys | 2.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Vetrichelvan Pugazendi

**Week**

2

**Timesheet period**

Period start 17/03/2021

Period end 12/03/2021

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 12/03/2021 | 13:00:00 | 01:30:00 | ANSYS | 12.50 |
| 13/03/2021 | 13:00:00 | 00:00:00 | Re-Mesh and Solve | 11.00 |
| 14/03/2021 | 12:00:00 | 06:00:00 | ANSYS | 18.00 |
| 15/03/2021 | 13:00:00 | | Try different Solvers | 0.00 |
| 16/03/2021 | | 08:00:00 | Try different Solvers | 0.00 |
| 17/03/2021 | 15:00:00 | 18:00:00 | Ansys | 3.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Vetrichelvan Pugazendi

**Week**

Enter week number 3

**Timesheet period**

Period start 18/03/2022

Period end 24/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 21/03/2022 | 11:50:00 | 21:30:00 | Stationary Golf Ball Simulation using K-Omega SST solver | 9.67 |
| 23/03/2022 | 14:00:00 | 14:30:00 | Design Domain for full Golf Ball | 0.50 |
| 23/03/2022 | 14:30:00 | 14:52:00 | Mesh | 0.37 |
| 23/03/2022 | 14:52:00 | 23:41:00 | Rotating golf ball simulation using RANS Solver at 2000 RPM | 8.82 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Vetrichelvan Pugazendi

**Week**

Enter week number 4

**Timesheet period**

Period start 24/03/2022

Period end 31/03/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 25/03/2022 | 10:50:00 | 21:30:00 | Rotating golf ball simulation using RANS Solver at 3000 RPM | 10.67 |
| 26/03/2022 | 14:00:00 | 14:30:00 | Rotating golf ball simulation using RANS Solver at 3000 RPM | 0.50 |
| 28/03/2022 | 12:00:00 | 15:40:00 | Re-Mesh using Inflation for validation and run simulation | 3.67 |
| 29/03/2022 | 11:00:00 | 22:21:00 | Try Different Solvers for validation | 11.35 |
| 30/03/2022 | 14:52:00 | | Re-Mesh using Inflation and sizing and run simulation | 0.00 |
| 31/03/2022 | | 18:10:00 | Try Different Solvers for validation | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Vetrichelvan Pugazendi

**Week**

Enter week number 5

**Timesheet period**

Period start 31/03/2022

Period end 07/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 31/03/2022 | 10:00:00 | 19:30:00 | Try Different Solvers for validation | 9.50 |
| 01/04/2022 | 14:00:00 | 18:10:00 | Try Different Solvers for validation | 4.17 |
| 03/04/2022 | 12:00:00 | 18:00:00 | Try Different Solvers for validation | 6.00 |
| 04/04/2022 | 15:00:00 | 22:00:00 | Refine mesh around golf ball and simulate | 7.00 |
| 06/04/2022 | 12:00:00 | | fixing issues with cl and cd values and simulating | 0.00 |
| 07/04/2022 | | 18:00:00 | experimenting different solvers | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Vetrichelvan Pugazendi

**Week**

Enter week number 6

**Timesheet period**

Period start 07/04/2022

Period end 14/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 09/04/2022 | 09:00:00 | | Run Simulation for 2000-3000 RPM for different angle of attacks | 0.00 |
| 10/04/2022 | | | Run Simulation for 2000-3000 RPM for different angle of attacks | 0.00 |
| 11/04/2022 | | | Run Simulation for 2000-3000 RPM for different angle of attacks | 0.00 |
| 12/04/2022 | | | Run Simulation for 2000-3000 RPM for different angle of attacks | 0.00 |
| 13/04/2022 | | | Run Simulation for 2000-3000 RPM for different angle of attacks | 0.00 |
| 14/04/2022 | | 19:40:00 | Run Simulation to validate CL and CD | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |


**Group details**

Group 4 | VvortexX

**Student details**

Vetrichelvan Pugazendi

**Week**

Enter week number 7

**Timesheet period**

Period start 14/04/2022

Period end 21/04/2022

| Date | Time Start | Time end | Activity Description | Hours worked |
|---|---|---|---|---|
| 15/04/2022 | 13:20:00 | | Ran Simulation for 2000-3000 RPM for different angle of attacks | 0.00 |
| 16/04/2022 | | | Ran Simulation for 2000-3000 RPM for different angle of attacks | 0.00 |
| 17/04/2022 | | | Ran Simulation for 2000-3000 RPM for different angle of attacks | 0.00 |
| 18/04/2022 | | 09:21:00 | Ran Simulation to validate CL and CD | 0.00 |
| 19/04/2022 | 10:00:00 | 01:00:00 | Created Presentation Template and added customs animations and formatted everyone's content | 15.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |
| | | | | 0.00 |

# N Minutes Documents

## 07/03/2022

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

**Supervision Meeting Notes**
**Group Project**

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 7th March 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Discussion of Tools Needed |
| 4. | Discussion of how we will structure future meetings |
| 5. | Explanation and discussion of simulation work |
| 6. | Any other businesses |
| 7. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**15:45** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X (Minutes writer) |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X (Meeting ruler) |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | None |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 2) Minutes (Done during meetings)

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | 15.46: Talk about our understanding of the subject in view of the comments made by the supervisors during the 1st presentation | |
| 3. | 16.00: Began discussion of tools | |
| 3. | 16.12: Began talks about CircleCI and code environments | |
| 3. | 16.20: Began talks about specific libraries and languages to be used | Continue research of tools and install tools and coding environments (**Everyone**) |
| 4. | 16.35: Began talks about future meeting structures and how we will separate meeting roles | Format documents for future meetings (**DS, PC**) |
| 5. | 16.37: **VP** began presentation of his current work (CFD implementation) | |
| 5. | 17.07: Began questions and discussion about the presentation | Begin the initial research stage. Research Vortices and CV/ML techniques (**Everyone**) |
| 6. | 18.00: None | |
| 7. | 18.05 Ended Discussion and decided next meeting time and date | 10th March at 15.10pm |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 10th March 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Project strategy/ plan via Jira |
| 4. | Share tasks amongst team members and set timelines |
| 5. | Discuss the architecture further focusing on the APIs |
| 6. | Any other businesses |
| 7. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**15:15** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X (Minutes writer) |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X (Meeting ruler) |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | None |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 2) Minutes (Done during meetings)

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | 15.17: Discussion on simulation work advancement | Ask the CFD lecturer how to automate timestep evolution on Crescent Fluent Simulations (**VP**) |
| 2. | 15.30: Discussion on possible ML/ CMV techniques | Start by a supervised approach before an unsupervised one (**AI**) |
| 3. | 16.10: Clarify usefulness of JIRA and relevance of project management tools | |
| 4. | 16.22: Debrief of everyone's work since the last meeting | Research on YOLO algorithm, Microservices, APIs, Different solvers for CFD (**Everyone**) |
| 5. | 16.35: Discuss the product architecture | Use Docker to containerize the application to ensure flexibility, sustainability of the software (**SE**) |
| 6. | 16.40: Hypothesis of how to increase the number of simulations | Vary parameters like the shape of the ball, vary the solvers, might vary simulation area (**VP**) |
| 6. | 17.07: Began questions and discussion about the presentation | Begin the initial research stage. Research Vortices and CV/ML techniques (**Everyone**) |
| 6. | 18.00: Labelling scheme, discussion about structuring input data | |
| 7. | 18.15 Ended Discussion and decided next meeting time and date | 14th March at 15.30pm |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 14th March 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Restrategise/ project tasks allocation |
| 4. | Any other businesses |
| 5. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**12:00** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X (Minutes writer) |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X (Meeting ruler) |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | None |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 2) Minutes (Done during meetings)

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | 12:05 We reviewed the actions from the last meeting. Vetri got an answer to his question on how to automate timestep evolution | Done |
| 2. | 12:07 Reflect on CNN (AI) | AI group needs to reflect on the CNN |
| 2. | 12:09 Vetri tried different solvers<br>• Large Eddy Simulations<br>• Scale Adaptive simulation<br>• Reynolds Average numerical simulation.<br>The last two solvers will be used | Extract plot from para view and send to AI team (VP) |
| 2. | 12:25<br>Yolo explanation by Daniel | Split in square: compare similarity to our dataset<br>Research on how to use python to code APIs (everyone)<br>Code architecture reflection (SE) |
| 3. | 12:39<br>Restrategise/ project tasks allocation | Market analysis, properly reset Jira, risks analysis, requirements, expectations (everyone) |
| 4. | 12:58<br>Other business: unsupervised ML possibilities | Discuss vortex core theory (VP, everyone) |
| 5. | 13:30<br>Ended discussion and decided next meeting date and time | 17th march at 11:00 |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 17th March 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Discussion of Tools Needed |
| 4. | Theory & AI Discussion |
| 5. | Any other businesses |
| 6. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**11:00** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X (Minutes writer) |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | Absent |
| **Tae Yong Kim (TK) - AI** | S332600 | X (Meeting ruler) |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| Itohanoghosa Eregie (Interview) | S356017 |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 2) Minutes (Done during meetings)

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | 11:02 CNN discussed | Find some references for using CNN (AI) |
| 2. | 11:11 Extraction of Plots from Paraview | Cropping vortex image regions and automatic generating vortex data (AI) |
| 2. | 11:23 API Investigation | Continue API Investigation (Everyone except CFD) |
| 2. | 11:25 Began discussing Market Analysis | Compare the different frameworks, Requirement, Expectation (SE)<br>Market analysis (AI, CFD) |
| 3. | 11:35 PC began presenting problems in JIRA | Reset properly JIRA (Everyone)<br>Discuss Agile Procedure (SE) |
| 4. | 11:55 Market Analysis on CFDs Solvers | |
| 4. | 12.17 Theory discussion | Research about theory from each perspective (Everyone) |
| 5. | 12:48 TK began presenting AI progress | Get some functioning codes (AI)<br>Share plt file (CFD, AI)<br>Investigate different solvers and compare experimental data (CFD) |
| 6. | 13.03 Ended Discussion and decided next meeting time and date | 21th March at 11.30pm |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 21st March 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Sprint 1: advancement discussion |
| 4. | Documentation |
| 5. | Any other businesses |
| 6. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**14:00** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X (Meeting ruler) |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X (Minutes writer) |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | None |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| **Action item** | **State** |
|---|---|
| Find some references for using CNN (AI) | Done (being improved & added) |
| Cropping vortex image regions and automatic generating vortex data (AI) | In progress |
| Continue API Investigation (Everyone except CFD) | Gained a general understanding of API development (waiting for the source code to Implement it) |
| Compare the different frameworks, Requirement, Expectation (SE) | In progress |
| Market analysis (AI, CFD) | In progress |
| Reset properly JIRA (Everyone) | Done |
| Research about theory from each perspective (Everyone) | In progress |
| Get some functioning codes (AI) | In progress (low accuracy working code) |
| Investigate different solvers and compare experimental data (CFD) | In progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings)

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | 14:00 We reviewed the actions from the last meeting and discussed our current pace. | ● Implement/adopt ml algo for current mockup(DW)<br>● Introduction to CFD for the report, Intro to SE, theory for CFD, Thoughts about the test cases(PC)<br>● Software Requirements documentation, build pipeline(IE)<br>● Get the simulation of spinning golf ball, Study the effect of inflation on cl and cd values(VP)<br>● Get trial code for pre-processing the image file for training data, specifically looking at generating vector data and using the FFT(DS)<br>● Study Q criterion for application in machine learning, Improve the model(TK)<br>● Get preliminary model for vortex detection, Research on generating artificial data(II) |
| 3. | 14:40 Discussion and understanding of neural networks. | Make sure every team member understands the basics of neural networks. |
| 4. | 15 :10 Individual tasks assignment for the upcoming week. | Build a cohesive list of tasks that can be equally distributed among group members for faster progression of the team. |
| 5. | 15:50<br>Discussion on Project Documentation | A skeleton report needs to be created that will help group members add content for the report simultaneously as they work on the practical aspects. |
| 6. | 16:30<br>Ended discussion and decided next meeting date and time | 24th march at 13:00 |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 24th March 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**13:35** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X (Meeting ruler) |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X (Minutes writer) |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| Implement/adopt ML algo for current SE-mockup<br>(AI) | In-progress<br>Creating and Checking Mock-up model |
| Introduction to CFD for the report, Intro to SE, theory for CFD, Thoughts about the test cases<br>(PC) | In-progress<br>(Finalizing Introduction and CFD theory) |
| Software Requirements documentation, build pipeline<br>(IE) | In-progress<br>(Testing dummy code / Dockerizing pipeline with front-end) |
| Get the simulation of spinning golf ball, Study the effect of inflation on cl and cd values<br>(VP) | Done (2,000 RPM)<br>In progress (3,000 RPM)<br>To discuss inflation with DR. Tom |
| Get trial code for pre-processing the image file for training data, specifically looking at generating vector data and using the FFT<br>(DS) | In progress<br>(Testing SIFT features and Vortex Gradients) |
| Study Q criterion for application in machine learning, Improve the model<br>(TK) | Done<br>(Implemented tutorial) |
| Get preliminary model for vortex detection, Research on generating artificial data<br>(II) | In progress<br>Creating and Checking Mock-up model |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | (13:40) Review and Discussion last meeting minutes | Work on skeleton report<br><br>(Everyone) |
| 2. | (13:45) Making and Applying deep learning model for Architecture<br><br>Discussion on applying API | Get preliminary model for vortex detection, Research on generating artificial data: Creating and Checking Mock-up AI model ready to be used by SEs<br><br>(AI, II) |
| 2. | (14:00) Reviewing test sample (2,000 RPM model) and Discussion | Introduction to CFD for the report, Intro to SE, theory for CFD, Thoughts about the test cases<br><br>(PC) |
| 2. | (14:20) Reporting software requirements documentation and pipeline with front-end | Software Requirements documentation, build pipeline<br><br>(IE) |
| 2. | (14:30) Reviewing CFDs Solvers theory | Get the simulation of spinning golf ball, Study the effect of inflation on cl and cd values: Testing SIFT features and Vortex Gradients<br><br>(VP) |
| 2. | (15:00) Explanation on test deep learning model | Implement/adopt ml algo for current mockup: Creating and Checking Mock-up model<br><br>(SE, DW) |
| 2. | (15:30) Review testing SIFT features and vortex gradients | Get trial code for pre-processing the image file for training data, specifically looking at generating vector data and using the FFT<br><br>(DS) |
| 3. | (15:45) Respectively manage JIRA | Respectively manage JIRA, properly reset advancements<br><br>(Everyone) |
| 4. | (16.26) Ended Discussion and decided next meeting time and date | 28th March at 11.30pm |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 28th March 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**12:12** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X (Meeting ruler) |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | Absent |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X (Minutes writer) |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | |

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| Work on skeleton report<br><br>(Everyone) | In-progress<br><br>Started |
| Get preliminary model for vortex detection, Research on generating artificial data: Creating and Checking Mock-up AI model ready to be used by SEs<br><br>(AI, II) | Done<br><br>(Need to be improved and dockerized) |
| Introduction to CFD for the report, Intro to SE, theory for CFD, Thoughts about the test cases<br><br>(PC) | In-progress |
| Software Requirements documentation, build pipeline<br><br>(IE) | In-progress |
| Get trial code for pre-processing the image file for training data, specifically looking at generating vector data and using the FFT: Testing SIFT features and Vortex Gradients<br><br>(DS) | Done |
| Get the simulation of spinning golf ball, Study the effect of inflation on cl and cd values<br><br>(VP) | In progress<br><br>(1 case done) |
| Implement/adopt ml algo for current mockup: Creating and Checking Mock-up model<br><br>(SE, DW) | In progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | (12:10) Review and Discussion last meeting minutes (Actions) | • Work on skeleton report (Everyone)<br>• Introduction to CFD for the report, Intro to SE, theory for CFD, Thoughts about the test cases (PC)<br>• Software Requirements documentation, build pipeline (IE)<br>• Get the simulation of spinning golf ball, Study the effect of inflation on cl and cd values (VP)<br>• Implement/adopt ml algo for current mockup: Creating and Checking Mock-up model (SE, DW)<br>• Incorporate image-proc into ML model (DS)<br>• Stabilise ML model (AI : II & TK) |
| 3. | (13:15) Respectively manage JIRA | Respectively manage JIRA, properly reset advancements<br>(Everyone) |
| 4. | (13.26) Ended Discussion and decided next meeting time and date | 21th March at 15.00pm |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 31st March 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**15:15** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X (Minutes writer) |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | Absent (Personal Reasons) |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X (Meeting ruler) |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | |

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| **Action item** | **State** |
|---|---|
| Work on skeleton report<br>(Everyone) | In-progress |
| Introduction to CFD for the report, Intro to SE, theory for CFD<br>(PC) | In-progress |
| Thoughts about the test cases<br>(PC) | In-progress |
| Software Requirements documentation, build pipeline<br>(IE) | In-progress |
| Get the simulation of spinning golf ball, Study the effect of inflation on cl and cd values<br>(VP) | In-progress<br>(Ran in steady-state, no experimental data) |
| Implement/adopt ml algo for current mockup: Creating and Checking Mock-up model<br>(SE, DW) | In-progress |
| Incorporate image-proc into ML model<br>(DS) | Delayed due to other tasks taking priority |
| Stabilise ML model<br>(AI : II & TK) | In-progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | Review and discussion of actions:<br>(15.15) Work on Skeleton Report | - Work on skeleton report (Everyone) |
| | (15:36) Introduction to CFD for the report, Intro to SE, theory for CFD + Thoughts about the test cases | - Introduction to CFD for the report, Intro to SE, theory for CFD (PC)<br>- Thoughts about the test cases (PC) |
| | (15:39) Software Requirements documentation, build pipeline | - Review Software Requirments and add in highlighted areas (Everyone) |
| | (15:52) Get the simulation of spinning golf ball, Study the effect of inflation on cl and cd values | - Remesh and discuss with Dr. Tom to improve accuracy of results (VP) |
| | (15:56) Implement/adopt ml algo for current mockup: Creating and Checking Mock-up model + Stabilise ML model | - Improve YOLO model and incorporate it into the API (TK, II, DW) |
| | (16:00) Incorporate image-proc into ML model | - Label data and perform ML on SIFT data using Neural Network |
| 3. | (16:10) Discussions of other potential simulation cases<br>(16:15) Discussion of next sprint goal | <No Action><br>- Add Jira actions to next sprint including test status (Everyone) |
| 4. | (16:25) End Meeting and decide next meeting time | 4th April 2022 15:00 |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 4th April 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**11:10** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X (Meeting ruler) |
| **Tae Yong Kim (TK) - AI** | S332600 | X (Minutes writer) |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| Work on skeleton report (Everyone) | In-progress |
| Introduction to CFD for the report, Intro to SE, theory for CFD (PC) | In-progress |
| Thoughts about the test cases (PC) | In-progress |
| Review Software Requirments and add in highlighted areas (Everyone) | In-progress |
| Remesh and discuss with Dr. Tom to improve accuracy of results (VP) | In-progress |
| Improve YOLO model and incorporate it into the API (TK, II, DW) | In-progress |
| Label data and perform ML on SIFT data using Neural Network | Done |
| Add Jira actions to next sprint including test status (Everyone) | Delayed due to other tasks taking priority |
| Stabilise ML model<br>(AI : II & TK) | In-progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | Review and discussion of actions:<br>(11.20) Work on Skeleton Report<br>(12:15) Discuss meeting minutes<br>(More detail state) | - Work on skeleton report (Everyone) |
| | (11:30) Discuss how to make paper (methodology (procedure part) : (what you did (flow chart..), research theory) | - Introduction to CFD for the report, Intro to SE, theory for CFD (PC) |
| | (11:45) Evaluate precision performance<br>Get test case template (Applying Training number / Dataset change)<br>(Applying Ball design change)<br>Review document of software requirement(deliverable, DB, (till 7th) | - Thoughts about the test cases (PC) |
| | (12:20) Share the plan (5th) for reviewing of Software Requirements documentation, build pipeline | - Review Software Requirments and add in highlighted areas (Everyone) |
| | (12:25) Study effect of mech refinement on CL & CD | - Remesh and discuss with Dr. Tom to improve accuracy of results (VP) |
| | (12:30) Get test case and Documentation (version update) | - Improve YOLO model and incorporate it into the API (TK, II, DW) |
| | (12:50) Label data and perform ML on SIFT data using Neural Network | - Augment current SIFT Neural Network to produce bounding boxes and use different image sizes (DS) |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

| | | |
|---|---|---|
| 3. | (12:55) User accuracy<br>(13:00) Discussion of next sprint goal | <No Action><br>- Add Jira actions including test status (Everyone) |
| 4. | (13:00) End Meeting and decide next meeting time | 7th April 2022 15:00 |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 7th April 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**13:10** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X (Minutes writer) |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X (Meeting ruler) |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

## 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| - Work on skeleton report (Everyone) | Done |
| | |
| - Introduction to CFD for the report, Intro to SE, theory for CFD (PC) | Done |
| - Thoughts about the test cases (PC) | In progress |
| - Review Software Requirments and add in highlighted areas (Everyone) | In progress |
| - Remesh and discuss with Dr. Tom to improve accuracy of results (VP) | In progress |
| - Improve YOLO model and incorporate it into the API (TK, II, DW)<br>- Augment current SIFT Neural Network to produce bounding boxes and use different image sizes (DS) | Improved but In progress<br>In progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | (13:45) review and discussions of actions : | (Everyone) Work on report |
| 2. | (13:50) Discussion on the presentation | (Everyone) Think what to put in presentation |
| 2. | (14:10) Software document review | - Degression on database and store informations like vortex report (DW)<br>- |
| 2. | (15:05) Remesh and discussion | (VP) Correction on CL |
| 2. | (15:10)YOLO accuracy | Improve YOLO model and incorporate it into the API (TK, II, DW) and testing model<br>-Link with AI (DS) |
| 3. | (15:15)Make user authentification | (DW) Create user authentification feature |
| 3. | (15:20) | (DS)Find core using gradient descent method |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 11th April 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**12:10** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X (Meeting ruler) |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X (Minutes writer) |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | |

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| (Everyone) Work on report | In progress |
| (Everyone) Think what to put in presentation | In progress |
| - Degression on database and store informations like vortex report (DW) | In progress |
| (VP) Correction on CL | Done |
| Improve YOLO model and incorporate it into the API (TK, II, DW) and testing model<br>-Link with AI (DS) | The Api is built for mobile net, the model load procedure is slightly different which can be replaced easily. |
| (DW) Create user authentification feature | In progress |
| (DS)Find core using gradient descent method | In progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | Poster Presentation (12:15) | Get Poster Requirements (everyone) |
| 2. | Presentation (12:30) | Build the presentation |
| 2. | CL-CD variables (12:50) | Run simulation for rest of the RPM.<br>Run the simulation for different methods<br>Smooth ball simulation. (VP) |
| | Find core (13:00) | Need improvement , more experiment , work with AI(DS) |
| 2. | AI Model build progression (13:10) | Incorporate custom post and pre processing into the model, tune the model with suitable hyperparameters. Test the completed model with a stock yolo model to check the improvements. |
| 2. | Deploying as a microservice(13:15) | Deploy the model as a service. |
| 2. | Test quality assurance document(13:25) | Create Test quality insurance document (PC) |
| 2. | Test summary(13:35) | (AI)create test summary document |
| 4. | Next meeting | 14th april |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 14th April 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**15:20** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X (Minutes writer) |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X (Meeting ruler) |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| None | |

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| Get Poster Requirements (everyone) | In progress<br>Requirements finished, begin creating poster (Everyone) |
| Build the presentation (everyone) | In progress<br>Continue building the presentation (Everyone) |
| Run simulation for rest of the RPM.<br>Run the simulation for different methods<br>Smooth ball simulation. (VP) | In progress<br>Finished with 2-4k RPMs.<br>Continue with validation<br>Run the simulation for different methods<br>Smooth ball simulation (VP) |
| Need improvement of Core Locating , more experiment , work with AI (DS) | In progress<br>Core locating has been improved,<br>Need to incorporate into YOLO outputs (DS) |
| Incorporate custom post and pre processing into the model, tune the model with suitable hyperparameters. Test the completed model with a stock yolo model to check the improvements. (AI) | In progress |
| Deploy the model as a service (DW) | In progress |
| Create Test quality insurance document (PC) | In progress |
| Create test summary document (AI) | In progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 2. | (15:20) Create Test quality insurance document (PC) and discussion of how quality insurance is broken down | Finish Quality Strategy document (PC) |
| 2. | (16:40) Get Poster Requirements (Everyone) | Develop poster contents (Everyone) |
| 2. | (17:20) Build the presentation (Everyone) | Continue building the presentation (Everyone) |
| 2. | (17:30) Run simulation for rest of the RPM.<br>Run the simulation for different methods<br>Smooth ball simulation. (VP) | Continue with validation<br>Run the simulation for different methods<br>Smooth ball simulation (VP) |
| 2. | (17:35) Need improvement of Core Locating , more experiment , work with AI (DS) | Need to incorporate core locating with YOLO outputs (DS) |
| 2. | (17:38) Incorporate custom post and pre processing into the model, tune the model with suitable hyperparameters. Test the completed model with a stock yolo model to check the improvements. (AI) | Incorporate custom post and pre processing into the model, tune the model with suitable hyperparameters. Test the completed model with a stock yolo model to check the improvements. (AI) |
| 2. | (17:40) Deploy the model as a service (DW) | Deploy the model as a service (DW) |
| 2. | (17:43) Create test summary document (AI) | Create test summary document (AI) |
| 3. | No Other Business | |
| 4. | (17:45) Next meeting | 18th April |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 18th April 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Presentation flow discussion |
| 4. | Poster design |
| 5. | Any other businesses |
| 6. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**14:20** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X(Minutes writer) |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X (Meeting ruler) |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| | |

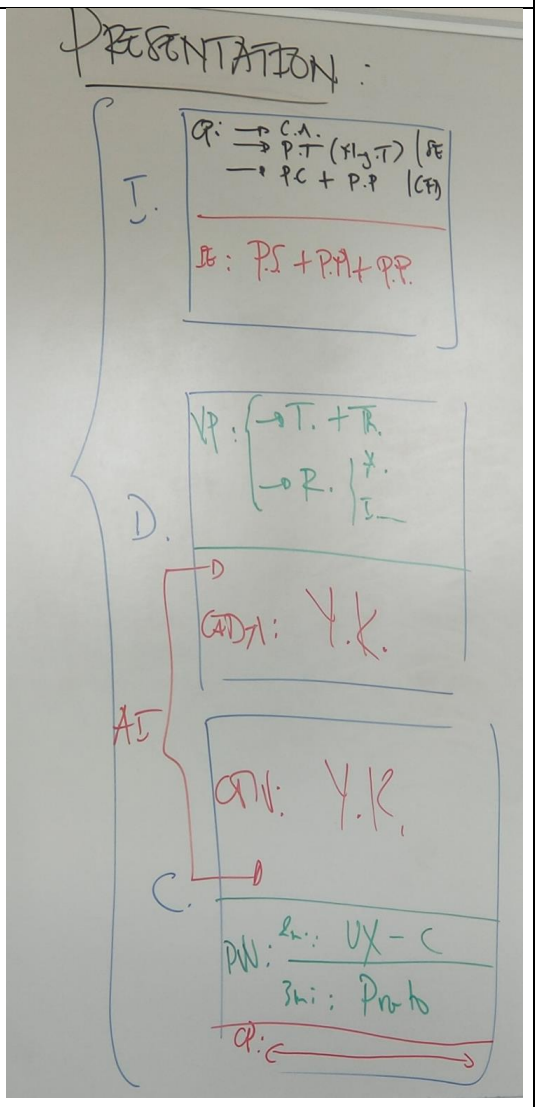
Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| Finish Quality Strategy document (PC) | Need test scheme |
| Develop poster contents (Everyone) | In progress |
| Continue building the presentation (Everyone) | In progress |
| Continue with validation<br>Run the simulation for different methods<br>Smooth ball simulation (VP) | Done |
| Need to incorporate core locating with YOLO outputs (DS) | Done |
| Incorporate custom post and pre processing into the model, tune the model with suitable hyperparameters. Test the completed model with a stock yolo model to check the improvements. (AI) | Done |
| Deploy the model as a service (DW) | In progress |
| Create test summary document (AI) | In progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 3. | (15:00) Presentation flow discussion |   |
| 4. | (16:15) Poster Design | |
| 5. | No other business | |
| 6. | (19:00) next meeting | 19th April 14:00 |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 19th April 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**14:50** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X(Minutes writer) |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X (Meeting ruler) |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| | |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| Finish Quality Strategy document (PC) | Need test scheme |
| Develop poster contents (Everyone) | In progress |
| Continue building the presentation (Everyone) | In progress |
| Deploy the model as a service (DW) | Done |
| Create test summary document (AI) | In progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 3. | (15:00) Poster Design | Assigned work |
| 3. | (17:00) Presentation | Assigned work |
| 4. | No other business | |
| 5. | (19:00) next meeting | 21 April 15:00 |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

# Supervision Meeting Notes
## Group Project

| | | | | | | |
|---|---|---|---|---|---|---|
| **Project Name** | **VvortexX** | | | | | |
| **Project Supervisors** | Seemal Asif, Jun Li, Irene Moulitsas, Tom Teschner | | | | | |
| **Date of Meeting** | 21th April 2022 | | | | | |
| **Meeting held by** | In person | ☒ | Telephone | ☐ | Virtual | ☐ |

### 1) Agenda (Done before meetings)

| Agenda item | Agenda subject |
|---|---|
| 1. | Apologies |
| 2. | Actions from the previous meeting |
| 3. | Any other businesses |
| 4. | Date of next meeting |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

**15:00** Meeting started

| **Attendees**<br>The following is a list of all attendees (i.e. students and project supervisor). | | |
|---|---|---|
| **Name** | **Student Number** (if applicable) | **Signature** |
| **Daniel Smythe (DS) - AI** | S374685 | X |
| **Vetrichelvan Pugazendi (VP) - CFD** | S351400 | X |
| **David Wang (DW) - SE** | S369748 | X(Minutes writer) |
| **Itohanoghosa Eregie (IE) - SE** | S356017 | X |
| **Tae Yong Kim (TK) - AI** | S332600 | X |
| **Imad Ui Islam (II) - AI** | S371584 | X |
| **Pierre Cisse (PC) - SE/ CFD** | S371584 | X (Meeting ruler) |

SE: Software Engineering sub-team; CFD: Computational Fluid Dynamics sub-team, AI: Artificial Intelligence sub-team.

| **Apologies were received from:** | |
|---|---|
| **Name** | **Student Number** |
| | |

Please save this document using the following naming convention: **Meeting date – GRP group name**
e.g. 16.03.2020 – GRP MAN 01

### 2) Actions from the previous meeting (Done during meetings)

| Action item | State |
|---|---|
| Finish Quality Strategy document (PC) | Need test scheme |
| Continue building the presentation (Everyone) | In progress |
| Create test summary document (AI) | In progress |

Please save this document using the following naming convention: **Meeting date – GRP group name** e.g. 16.03.2020 – GRP MAN 01

### 3) Minutes (Done during meetings (New Actions for next))

| Agenda item | Point discussed | Action |
|---|---|---|
| 3. | (15:00) Presentation | (everyone)Assigned Tasks |
| 3. | (16:00) Presentation | Setting Deadlines |
| 4. | (17:00) Next meeting | 25 April 11:00 |

IMPORTANT Upon finalising the notes and actions, and once initialled by all participants please forward to SATMTaughtMeetingNotes@cranfield.ac.uk copying in your group project supervisor

# References


Aoki, K., Muto, K., & Okanaga, H. (2010). Aerodynamic characteristics and flow pattern of a golf ball with rotation. *Procedia Engineering*, *2*(2), 2431–2436.

De Gregorio, F., Coletta, M., Visingardi, A., & Iuso, G. (2019). *An assessment of vortex detection criteria for piv data in rotorcraft applications*.

Donahue, M. z. (2019). What is turbulence—and how can you calm down about it? *National Geographic*. https://www.nationalgeographic.com/travel/article/what-is-turbulence-explained

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Softmax units for multinoulli output distributions. MIT Press.

Hayashida, H., & Iwasa, Y. (1990). Aerodynamic shape effects of tall building for vortex induced vibration. *Journal of Wind Engineering and Industrial Aerodynamics*, *33*(1), 237–242.

Khosronejad, A., Kang, S., Wermelinger, F., Koumoutsakos, P., & Sotiropoulos, F. (2021). A computational study of expiratory particle transport and vortex dynamics during breathing with and without face masks. *Physics of Fluids*, *33*(6), 066605.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations*.

Lemaréchal, C. (2012). Cauchy and the gradient method. *Math Extra*, 251–254.

Liang, D., Wang, Y., Liu, Y., Wang, F., Li, S., & Liu, J. (2018). A cnn-based vortex identification method. *Journal of Visualization*, *22*.

Liu, Y., Sun, P., Wergeles, N., & Shang, Y. (2021). A survey and performance evaluation of deep learning methods for small object detection. *Expert Systems with Applications*, *172*, 114602.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60(2)*, 91–110.

Murphy, K., & Dainty, C. (2012). Comparison of optical vortex detection methods for use with a shack-hartmann wavefront sensor. *Optics express*, *20*, 4988–5002.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th International Conference on International Conference on Machine Learning*, 807–814.

Nguyen, N.-D., Do, T., Duc, T., & Le, D.-D. (2020). An evaluation of deep learning methods for small object detection. *Journal of Electrical and Computer Engineering*, *2020*, 1–18.

Pham, P., Nguyen, D., Do, T., Duc, T., & Le, D.-D. (2017). Evaluation of deep models for real-time small object detection, 516–526.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788.

Ren, S., He, K., Girshick, R. B., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *CoRR*.

Sobel, I. (2014). An isotropic 3x3 image gradient operator. *Presentation at Stanford A.I. Project 1968*.

Szeliski, R. (2022). *Computer vision: Algorithms and applications 2nd edition* (pp. 119–124, 127–129, 132–133). Springer.

Verma, S., Novati, G., & Koumoutsakos, P. (2018). Efficient collective swimming by harnessing vortices through deep reinforcement learning.

Yanyang, L., Shao, Y., Hongyu, C., Bin, W., Mingqi, H., & Rao, Y. (2020). Cnn-based blade tip vortex region detection in flow field, 46.

Zeiler, M. D., & Fergus, R. (2013). Visualizing and understanding convolutional networks. *CoRR*.