COMPUTATIONAL FLUID DYNAMICS FOR AUTONOMOUS UNDERWATER VEHICLES

JONATHAN SMITH
MICHAEL HINCHEY

ABSTRACT

This paper shows that Computational Fluid Dynamics or CFD can be used to study the motions of Autonomous Underwater Vehicles or AUVs that are being driven by their own spinning propellers. Results for an AUV undergoing depth and yaw maneuvers are presented. The AUV has one set of props for depth control and another set of props for yaw control. It also has a buoyancy tank to fine tune depth control. The motions of the AUV look quite realistic. The work seems to be unique.

BACKGROUND

We are developing a small autonomous underwater vehicle or AUV for survey type missions. It is ballasted such that it sits vertically in the water and has ability to yaw but its pitch and roll are insignificant. Its Degrees of Freedom or DOFs are basically uncoupled, which makes control simpler. It uses two vertical axis props for depth control and two horizontal axis props for yaw control. The props of the AUV are large and slowly rotating. The AUV also uses a small buoyancy tank to make sure weight and buoyancy are balanced. This paper explores the use of Computational Fluid Dynamics or CFD for studying the motion of the AUV.

Simulations of the motions of Autonomous Underwater Vehicles or AUVs often make use of data from steady state experiments. However, when vehicles are homing in on a rest state, the flows around them are quite complex with the AUV moving into and out of its own wake. These flows are definitely not steady, so use of steady state data is not justified. A review of the literature shows that others have used CFD to get the resistance characteristics of torpedo shaped AUVs. Some use an actuator disk to model the prop. None of this work is relevant to the present work. It seems that no one has used CFD to model a box shaped AUV that is being driven by its own spinning propellers. The present work shows that Computational Fluid Dynamics or CFD can model this case. It is based on the commercial CFD package FLOW 3D [1]. The intended audience is not developers of CFD codes but is instead developers of AUVs. Details of the CFD such as the governing equations can be found elsewhere, but the intended audience may not be familiar with these details so for completeness they are given in the paper.

COMPUTATIONAL FLUID DYNAMICS

In FLOW 3D, the flow field is discretized by an xyz system of grid lines. Small volumes or cells surround points where grid lines cross. FLOW 3D approximates each of the governing partial differential equations within each cell. Conservation of momentum considerations give:

$$\rho \left(\partial U/\partial t + U\partial U/\partial x + V\partial U/\partial y + W\partial U/\partial z \right)$$
$$+ A = - \partial P/\partial x$$
$$+ \left[\partial/\partial x (\mu\partial U/\partial x) + \partial/\partial y (\mu\partial U/\partial y) + \partial/\partial z (\mu\partial U/\partial z) \right]$$

$$\rho \left(\partial V/\partial t + U\partial V/\partial x + V\partial V/\partial y + W\partial V/\partial z \right)$$
$$+ B = - \partial P/\partial y$$
$$+ \left[\partial/\partial x (\mu\partial V/\partial x) + \partial/\partial y (\mu\partial V/\partial y) + \partial/\partial z (\mu\partial V/\partial z) \right]$$
$$\rho \left(\partial W/\partial t + U\partial W/\partial x + V\partial W/\partial y + W\partial W/\partial z \right)$$
$$+ C = - \partial P/\partial z - \rho g$$
$$+ \left[\partial/\partial x (\mu\partial W/\partial x) + \partial/\partial y (\mu\partial W/\partial y) + \partial/\partial z (\mu\partial W/\partial z)\right]$$

where U V W are the velocity components in the x y z directions, P is pressure, ρ is the density of water and μ is its effective viscosity.

Conservation of mass considerations give:

$$\partial P/\partial t + \rho c^2 \left(\partial U/\partial x + \partial V/\partial y + \partial W/\partial z \right) = 0$$

Although water is basically incompressible, for mass conservation, FLOW 3D takes it to be compressible. Mass is used to adjust pressure at points in the grid when the divergence of the velocity vector is not zero, in other words, when mass seems to be flowing into or out of a point in space.

For an AUV operating close to the water surface, a special function F, known as the Volume of Fluid or VOF function, can be used to locate the water surface. For water, F is taken to be unity: for air, it is taken to be zero. Regions with F between unity and zero must contain the water surface. Material volume considerations give for F:

$$\partial F/\partial t + U\partial F/\partial x + V\partial F/\partial y + W\partial F/\partial z = 0$$

Hydrodynamics flows are generally turbulent and contain many small eddies which move around in an erratic way. Like the molecules in a gas, the erratic motion of eddies in a flow tend to diffuse observable momentum. Engineers are usually not interested in the details of the eddy motion in a turbulent flow. Instead they need models which account for its diffusive character, because it can change the observable motion of a flow and the loads on bodies in the flow. These can be obtained from the momentum equations through a complex time averaging process. The time averaging process introduces source like terms A B C into the momentum equations. Each is a complex function of velocity and viscosity gradients:

$$A = \partial \mu/\partial y\ \partial V/\partial x - \partial \mu/\partial x\ \partial V/\partial y$$
$$+ \partial \mu/\partial z\ \partial W/\partial x - \partial \mu/\partial x\ \partial W/\partial z$$

$$B = \partial \mu/\partial x\ \partial U/\partial y - \partial \mu/\partial y\ \partial U/\partial x$$
$$+ \partial \mu/\partial z\ \partial W/\partial y - \partial \mu/\partial y\ \partial W/\partial z$$

$$C = \partial\mu/\partial y\ \partial V/\partial z - \partial\mu/\partial z\ \partial V/\partial y$$
$$+ \partial\mu/\partial x\ \partial U/\partial z - \partial\mu/\partial z\ \partial U/\partial x$$

The turbulence models produced by time averaging are known as eddy viscosity models. A popular eddy viscosity model found in FLOW 3D is known as the k-ε model, where k is the local kinetic energy of turbulence and ε is its local dissipation rate. Its governing equations are:

$$\partial k/\partial t + U\partial k/\partial x + V\partial k/\partial y + W\partial k/\partial z = T_P - T_D$$
$$+ [\ \partial/\partial x\ (\alpha\ \partial k/\partial x) + \partial/\partial y\ (\alpha\ \partial k/\partial y) + \partial/\partial z\ (\alpha \partial k/\partial z)\ ]$$

$$\partial\varepsilon/\partial t + U\partial\varepsilon/\partial x + V\partial\varepsilon/\partial y + W\partial\varepsilon/\partial z = D_P - D_D$$
$$+ [\ \partial/\partial x\ (\beta\partial\varepsilon/\partial x) + \partial/\partial y\ (\beta\partial\varepsilon/\partial y) + \partial/\partial z\ (\beta\partial\varepsilon/\partial z)\ ]$$

where

$$T_P = G\ \mu_t / \rho \qquad D_P = T_P\ C_1\ \varepsilon / k$$
$$T_D = C_D\ \varepsilon \qquad D_D = C_2\ \varepsilon^2 / k$$
$$\mu_t = C_3\ k^2 / \varepsilon \qquad \mu = \mu_t + \mu_l$$
$$\alpha = \mu/a \qquad \beta = \mu/b$$

where

$$G = 2\ [\ (\partial U/\partial x)^2 + (\partial V/\partial y)^2 + (\partial W/\partial z)^2\ ]$$
$$+ [\ \partial U/\partial y + \partial V/\partial x\ ]^2 + [\ \partial U/\partial z + \partial W/\partial x\ ]^2$$
$$+ [\ \partial W/\partial y + \partial V/\partial z\ ]^2$$

where C_D, C_1, C_2, C_3, a and b are constants based on data from simple experiments, μ_l is the laminar viscosity, μ_t is extra viscosity due to eddy motion, T_P and D_P are turbulence production functions and T_D and D_D are turbulence decay functions. The k-ε equations account for the convection, diffusion, production and decay of turbulence.

A unique feature of FLOW 3D, known as the General Moving Object or GMO, allows bodies to move through the grid. The motions of the bodies can be prescribed, or they can be coupled to the motion of the fluid. It allows for extremely complicated motions and flows. One can think of a GMO as a bubble in a flow, where the pressure on the inside surface of the bubble is adjusted in such a way that its boundary matches the shape of a body. FLOW 3D uses a complex interpolation scheme to fit the body into the grid.

For CFD, each governing equation is put into the form:

$$\partial M/\partial t = N$$

At points within the CFD grid, each governing equation is integrated numerically across a time step to get:

$$M(t+\Delta t) = M(t) + \Delta t\ N(t)$$

where the various derivatives in N are discretized using finite difference approximations. The discretization gives algebraic equations for the scalars P F k ε at points where grid lines cross and algebraic equations for the velocity components U V W at staggered positions between the grid points. Central differences are used to discretize the viscous terms in the momentum and turbulence equations. To ensure numerical stability, a mix of central and upwind differences is used for the convective terms. Collocation or lumping is used for source terms. To march the unknowns forward in time, the momentum equations are used to update U V W, the mass equation is used to update P and correct U V W, the VOF equation is used to locate the water surface and the turbulence equations are used to update k and ε.

The Semi Implicit Method for Pressure Linked Equations or SIMPLE procedure is used to get pressure through an iterative process involving mass and momentum. The SIMPLE procedure first uses the momentum equations to calculate velocities in the flow field and then plugs those velocities into a pressure adjustment equation based on the mass equation. If the divergence of the velocity field is positive, it means that there is a net flow out of the region and suction is needed to reduce that. If the divergence of the velocity field is negative, it means there is a net flow into the region and surge pressure is needed to reduce that. The SIMPLE procedure gives these suction and surge pressure adjustments and iterates until divergence is close to zero.

Special wall functions are used to skip over the sharp normal gradients in velocity and turbulence near walls. With these wall functions, boundary conditions are applied just outside the boundary layer next to the wall and not at the wall itself. This reduces computational time.

AUV GEOMETRY

The main hull of the AUV consisted of a tube with conical end caps. The AUV had two props for yaw control and two props for depth control. The AUV also used a buoyancy tank to fine tune depth control. The buoyancy tank was positioned on top of the hull. It consisted of a piston free to move vertically inside a cylinder, closed at the bottom and open at the top. The DOFs of the AUV are basically uncoupled which makes control simpler. The AUV is shown positioned in the CFD grid in Figure 1.

The AUV was small and moved in a water tank that was 1m deep and had 1m square cross section. The water depth was 0.87m. The AUV hull had a diameter of 0.089m. The diameter of each prop was 0.108m. Each prop had 3 blades. The distance out from the hull to the CG of each prop was 0.1m. The buoyancy tank had a diameter of 0.0508m and a height of 0.0635m. The piston in the tank had a diameter of 0.0254m and a height of 0.0127m. There was a wall boundary condition at the bottom of the water tank and a constant pressure over the water surface at the top. The sides were open boundaries that fluid could move across. The available computer power was limited so we had to use a coarse grid in this work. There were 175 cells in x direction, 175 cells in y direction and 190 cells in z direction. The cell size in each direction near the AUV was 0.004m.

Several different prop designs were studied. Each had a shroud to minimize the influence of tip vortices. The blades used in the work presented here are shown in Figure 2. The blades are symmetric, so they work the same forward and reverse. Also, there is a set of blades for heave and yaw. Each set consists of a blade and its mirror image. When one is rotating CW at a certain speed and the other is rotating CCW at the same speed, their thrusts add up but their torques cancel. As can be seen in Figure 2, the blades are quite thick. This was to ensure they could be seen by a coarse CFD grid. In our work on marine current energy devices [2,3], we found that thick blades gave results close to experimental data from thin blades. So, we expected that thick blades would work here. We had about 1.5 cells across a blade section. We found that flow leaked through the blades when there was less than 1 cell across a section. Note that, in the simulation, there were no physical connections between the props and the hull of the AUV. Instead, the connections between the props and the hull were virtual. We could add physical connections between the props and the hull, but they would not be seen by a coarse grid.

AUV SIMULATION

The simulations were performed on a DELL STUDIO XPS 9100 Computer. It has 12 GB of RAM and 918 GB of ROM. It has 8 CPUs, and its operating speed is 2.8 GHz. This is a good computer. But computation time is long for fine grids, so we had to use coarse grids. More processors would greatly speed up the computations for fine grids. This is so because of the parallel nature of the CFD computations. With explicit time stepping, each point in the grid can be updated in a random order. The cases reported here used a coarse grid and typically took about a day to run. With a grid twice as fine in each direction, cases would take approximately 16 days to run. There would be 8 times as many grid points and roughly twice the number of time steps. For exploratory work, it does not make sense to wait 16 days to see if something is working or not, so we used the coarse grid. The results obtained with the coarse grid physically made sense.

For the present work, the components of the AUVs were first drawn in the CAD software SOLID WORKS. The geometry files were imported into FLOW 3D as STL files. FLOW 3D has subroutines that allow the user to customize the simulation. Specifically, for the present work, this feature allowed us to determine the fluid loads on the spinning props of an AUV and to transfer those loads to the hull of the AUV. It also allowed us to spin the props using a control signal and make them move with the AUV. The subroutines are written in FORTRAN and were modified using the compiler Intel Fortran Composer XE for Microsoft Windows Studio. Any FLOW 3D user wishing to obtain the modified subroutines used in this work can download them from the folder AUV on the web page: www.engr.mun.ca/~mhinchey.

In FLOW 3D, the hull and the props were modelled as separate components free to move relative to each other. The loads on the spinning props were calculated by the customization subroutine **mvbfrc1**. It called subroutine **mvbprfrc**, which calculated the pressure loads, and subroutine **mvbshrfrc**, which calculated the viscous traction loads. These loads were made control loads for the hull. The hull was a coupled motion body in FLOW 3D, which means its equations of motion were solved. The spinning props were prescribed motion bodies in FLOW 3D, which means their inertias were ignored. For this reason, the mass of each prop was added to the mass of the hull, because it is, in reality, physically connected to the hull. The rotational inertia of the hull was adjusted to account for the motion of the center of gravity of each prop about the vertical axis of the hull. Customization subroutine **mvbvel** was used to make the props move as if they were physically attached to the hull. A translational velocity component of a prop consists of the corresponding translational velocity component of the hull plus a swing component due to its rigid body rotation around the vertical or z axis of the hull. The yaw angle of the hull was needed to break the swing component into x and y components. The z axis swing component is zero. In FLOW 3D, the spin of a prop around its own axis has to be set in its local body space coordinate system. The components of the spin were first calculated in the global space coordinate system and then transferred to the local body space coordinate system using the function **trfmvb**, which is available in the subroutine **mvbvel**.

Often, because of numerical stability issues, a time cycle must be repeated at smaller or larger time steps. We want to update the rotational velocity of each prop only after a proper time step is selected, and the time cycle is actually completed. Also, for certain geometry and load calculations, we need the yaw angle of the hull at the end of each time step. The yaw rate is known at the beginning of each time step. A simple Euler integration of yaw rate gives the yaw angle. Customization subroutine **qsadd** is called only at the end of a time cycle, when a step in time is actually completed.

We used it in our work to calculate the rotational velocity of the props and the yaw angle of the hull at the end of each time step.

FLOW 3D uses COMDECK files to store and transfer data from one subroutine to another. These are like the COMMON statements used in old FORTRAN codes. Certain variables in the customization subroutines had to be stored at the end of each time cycle. The COMDECK file **dumn** was used for this purpose. Control data also had to be supplied to the subroutines. The COMDECK file **cbusr** was used for this purpose. Data in this file was given at the end of the **prepin** file for the simulation. It contains all the data for the simulation. Once files were modified, they were compiled into the main FLOW 3D code using a BUILD function.

A first order model can account roughly for the rotational inertia of the props and the unsteady hydrodynamics of the blades. A first order model was used to change the rotational speed of the depth props based on a calculated depth error control signal. A first order model was also used to make the rotational speed of the yaw props home in on a constant command speed. These first order models are:

$$A\, dD/dt + B\, D = Q \qquad\qquad A\, dY/dt + B\, Y = P$$

$$Q = K_P\, E + K_I \int E d\tau + K_D\, dE/dt$$

In these equations, D is the rotational velocity of the depth props, Y is the rotational velocity of the yaw props, Q is the command speed of the depth props, P is the command speed of the yaw props and E is depth error. A simple Euler integration applied to these equations gave

$$D_{NEW} = D_{OLD} + \Delta t\,(Q_{OLD} - B\,D_{OLD})/A$$

$$Y_{NEW} = Y_{OLD} + \Delta t\,(P_{OLD} - B\,Y_{OLD})/A$$

where Δt is the time step. The control signal Q became

$$Q_{OLD} = K_P\,E_{OLD} + K_I \sum E_{OLD}\,\Delta t + K_D\,\Delta E_{OLD}/\Delta t$$

Euler integration gave for the yaw angle of the hull

$$H_{NEW} = H_{OLD} + \Delta t\,dH/dt$$

where H is yaw angle and dH/dt is yaw rate.

SIMULATION RESULTS

Because this work was exploratory in nature, only a few results are presented here. Some results for control of the AUV using depth/yaw props are shown in Figure 3. For these results, the piston in the buoyancy tank was in a neutrally buoyant position. The AUV started at the water surface and was commanded to move to a depth of 0.57m. At the start, the depth props were completely out of the water and the yaw props were partially submerged. Both starting spinning as soon as the mission started. The depth controller gains were K_P=250, K_I=250, K_D=25. No attempt was made to optimize these gains. A saturation limit was imposed on the depth control signal of 62.8 rad/s or 10 cps or 600 RPM. The command yaw spin speed was 6.28 rad/s or 60 RPM. The first order model parameters were A=0.1 and B=1.0. This means the response time of the props was around 0.1s, which is reasonable for a small AUV. The blue curve in Figure 3 shows the depth response of the AUV, while the green curve shows the spin of one of the depth props. The AUV can be seen to home in on the command depth, and the props can be seen to reverse direction several times to make this happen. Figure 4 shows some results for the case where the buoyancy tank was used to control depth. For these results, the spin of the depth props was zero. The green curve in Figure 4 shows the depth response for the case where the piston stayed at the bottom of the buoyancy tank during the run. In this case, buoyancy was less than weight and the AUV kept on moving downward. The blue curve in Figure 4 shows the depth response for the case where the piston started at the bottom of the buoyancy tank but moved to the top of the tank once the AUV crossed the command depth. In this case, buoyancy became greater than weight, which caused the AUV to gradually stop and move back up to the water surface. Obviously, these make sense physically. Figure 5 shows a screen shot of the prop AUV just after it went below the water surface.

CONCLUSIONS AND FUTURE WORK

This paper showed that Computational Fluid Dynamics can be used to study motion control of Autonomous Underwater Vehicles being driven by their own spinning propellers. The work was exploratory in nature. To extend it, we need more computer power, so we can use finer grids for accuracy and create a larger workspace for the AUV to move around in. It will also be necessary to compare CFD results to experimental data. For this purpose, a physical model of the AUV is being constructed.

REFERENCES

[1] Flow Sciences (2014): FLOW 3D CFD: User Manual.

[2] Nahidul Khan, Jonathan Smith and Michael Hinchey (2013): "A CFD Study of a Savonius Water Current Turbine", JOT, Vol 8, No 1.

[3] Nahidul Khan (2014): "Development of a Novel Kaplan Marine Current Energy Device", Faculty of Engineering, Memorial University, PhD Thesis.

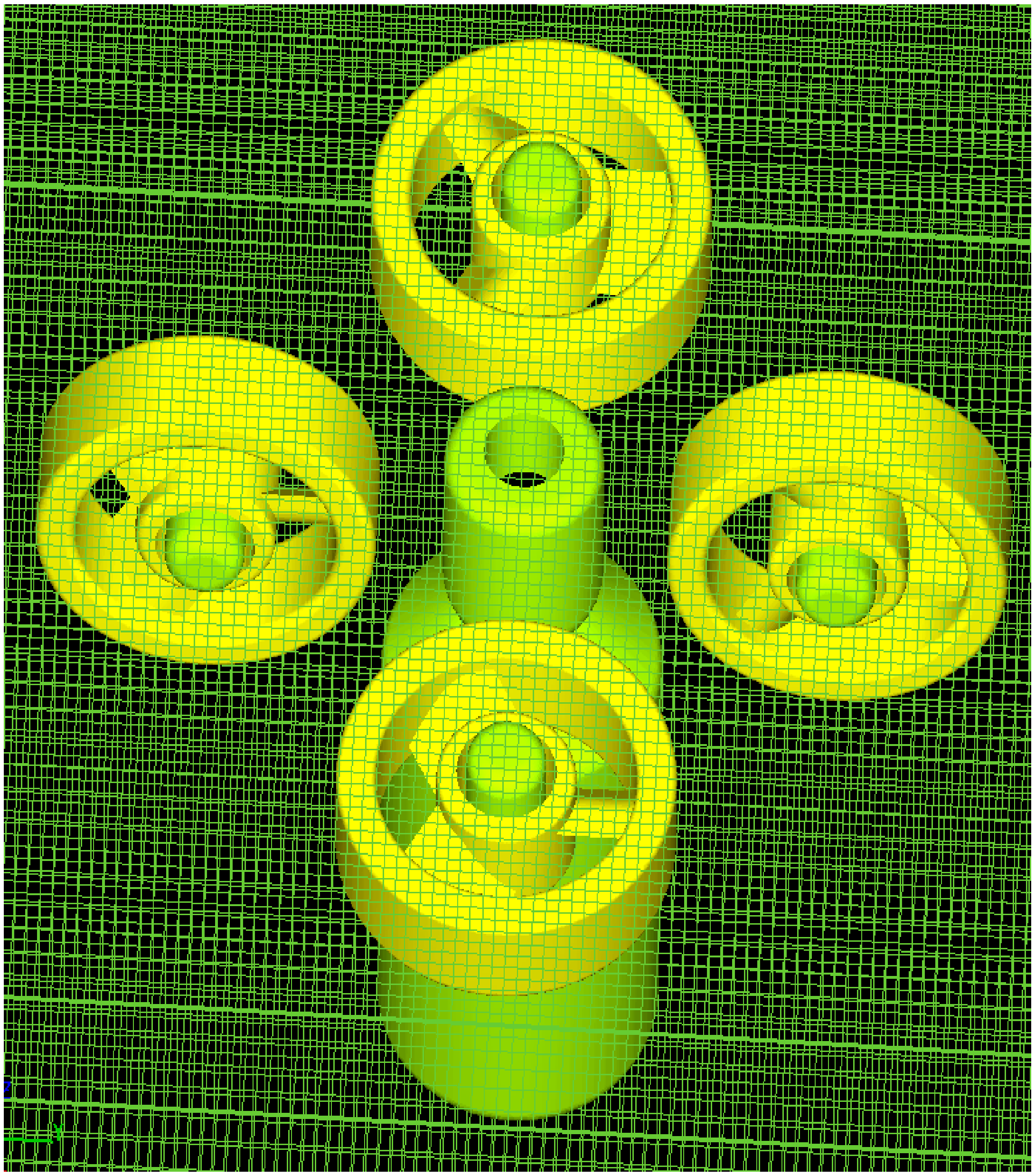

Figure 1: AUV Geometry

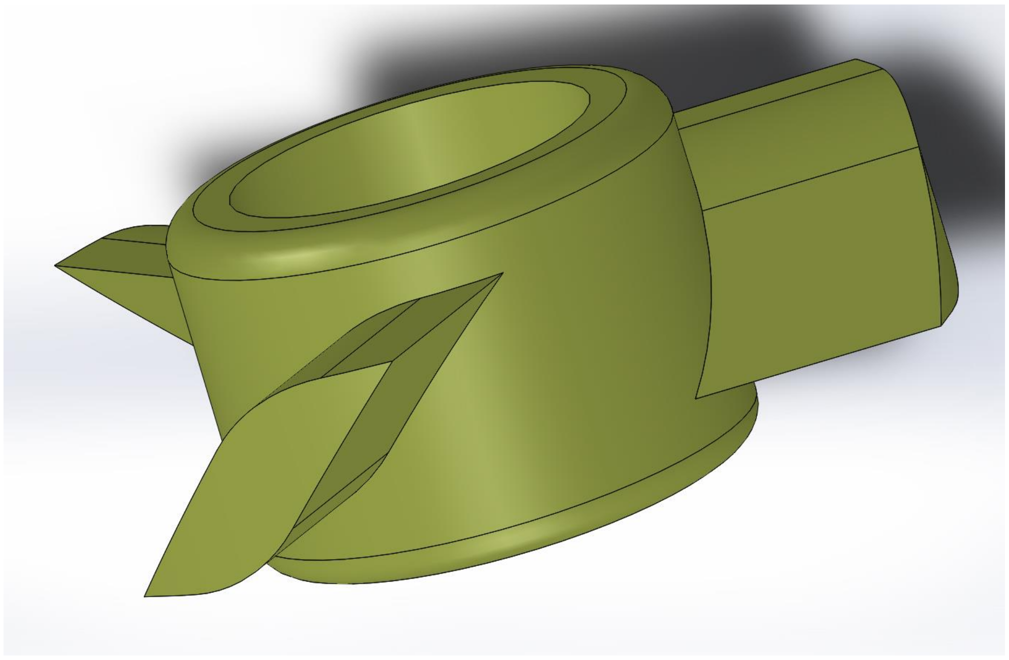

Figure 2: Propeller Blade Profile

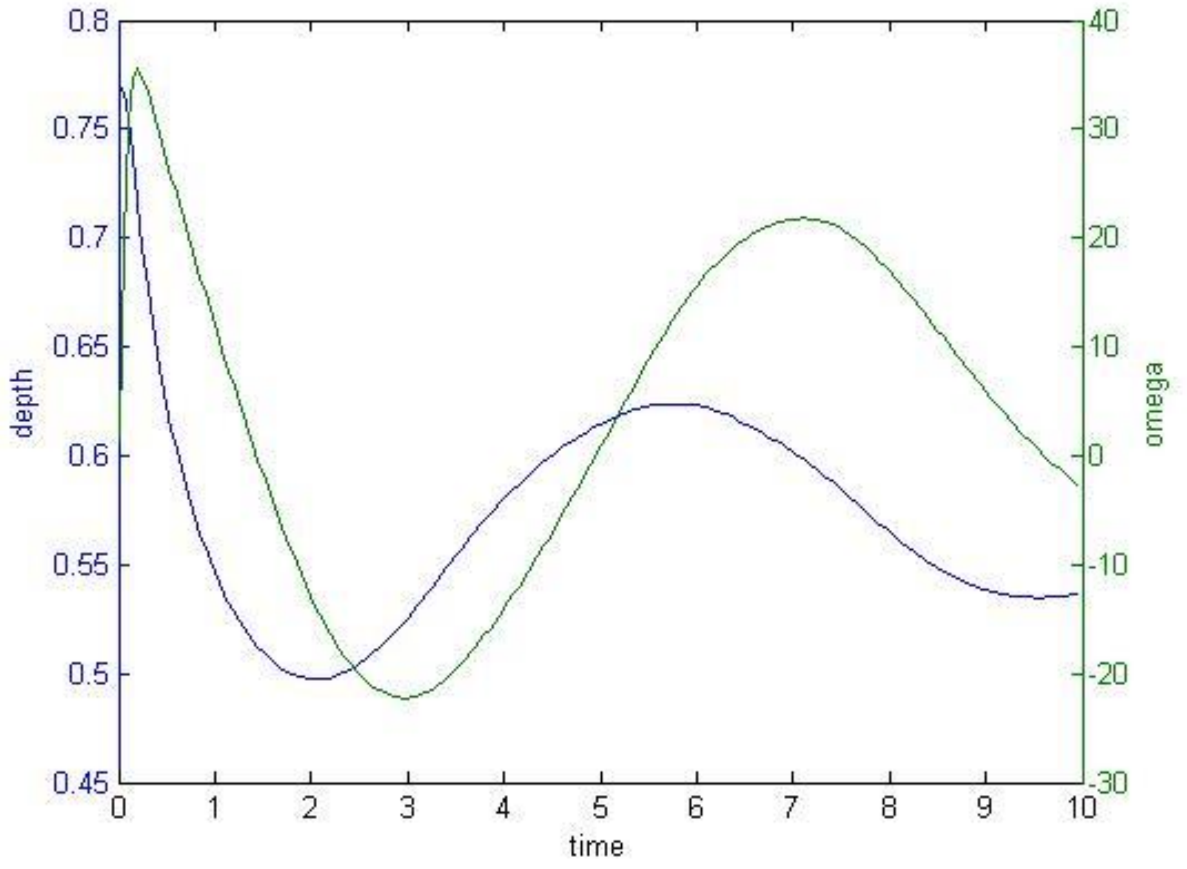

Figure 3: Response of Prop AUV

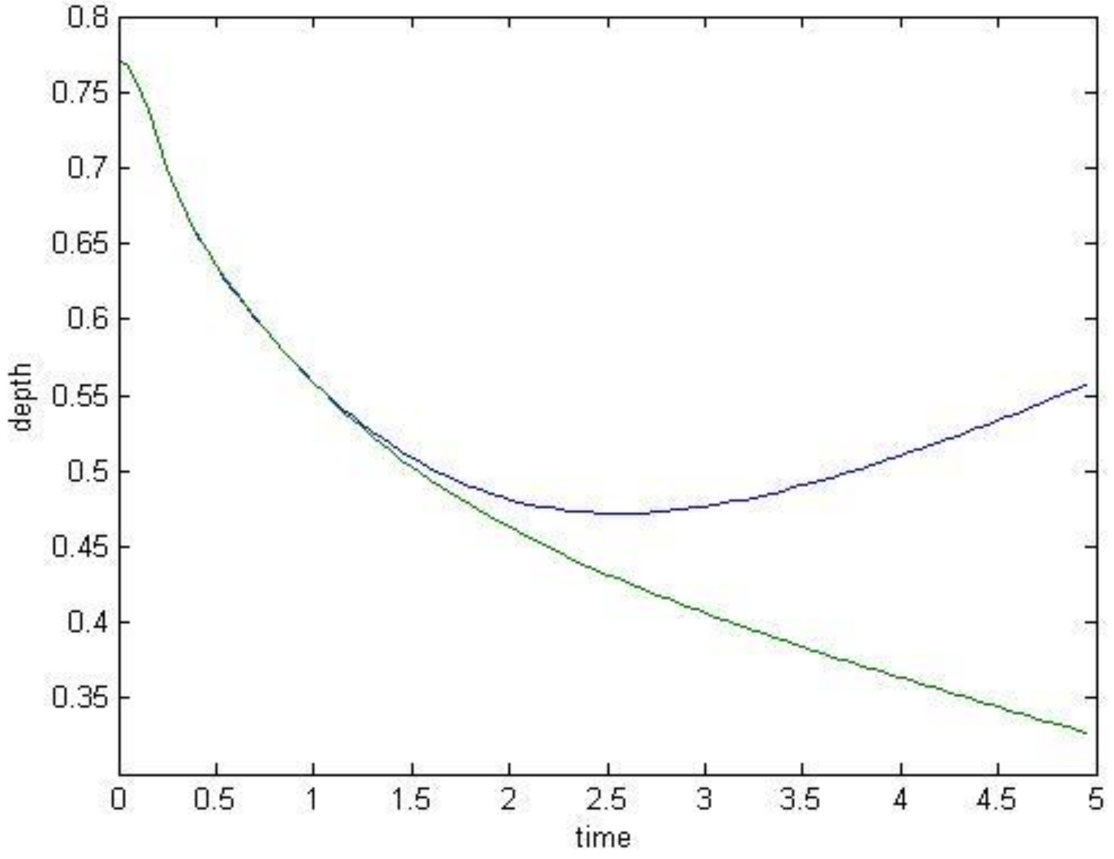

Figure 4: Response of Tank AUV

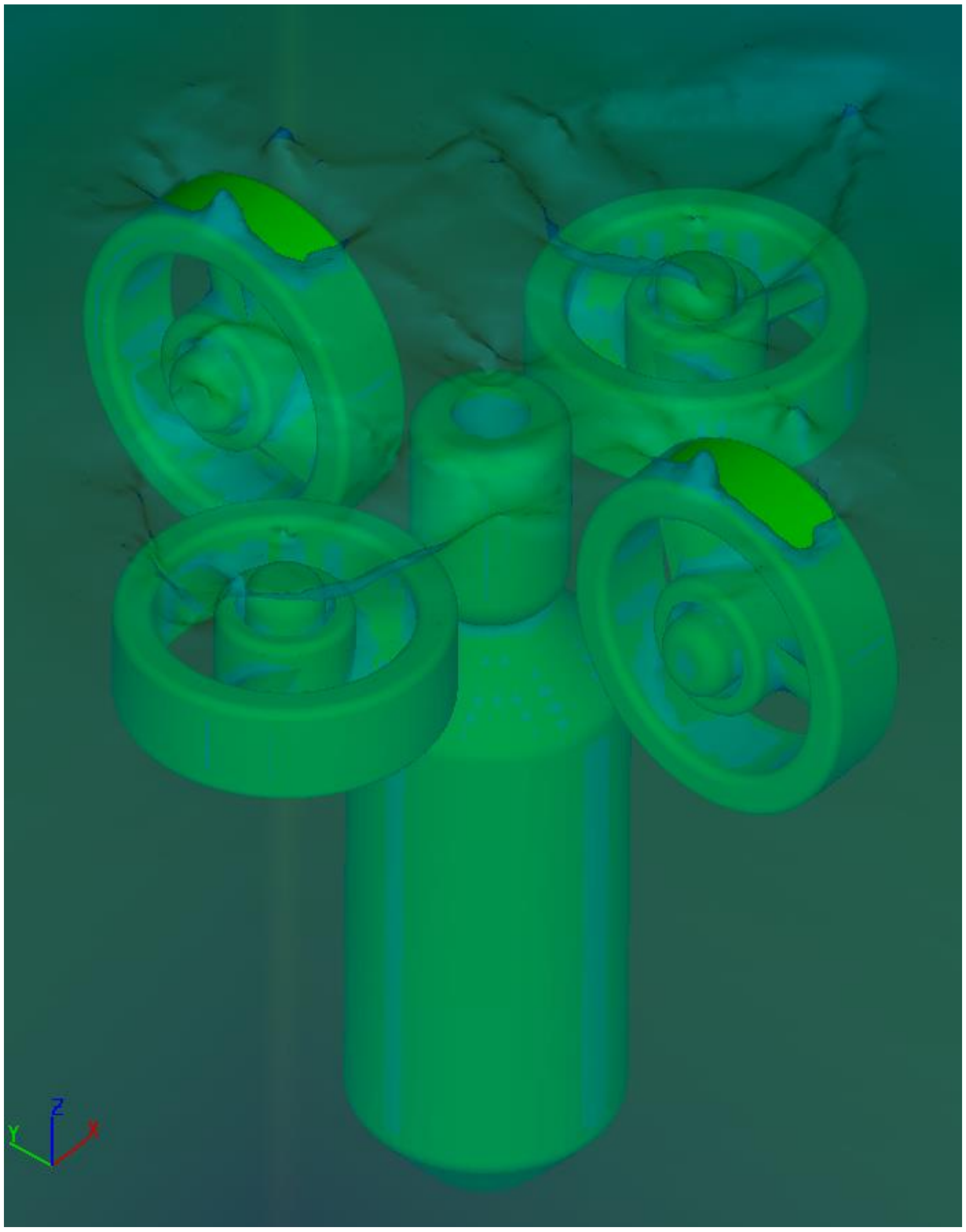

Figure 5: Screen Shot of AUV