Proposed configuration for MIT ROV, featuring four thrusters with contrarotating propellers.

The original impetus of this research project is to design a set of small diameter, contrarotating propellers to be used by the MIT ROV Team on their entry into the annual MATE ROV Competition. Since it started, the scope of the project has expanded to utilizing existing computational tools and producing additional tools for more general thruster design for marine robots.

Circulation Optimization

I’m using Jake Kerwin’s Propeller Lifting Line (PLL) code as the base of my design. It optimizes the circulation distribution for a propulsor with one or more elements, each of which can be a propeller, stator, or contrarotating propeller. The program requires inputs of inflow velocity ($U_{in}$), fluid density ($\rho$), number of elements, element position, number of blades (Z), element diameter (D), rotational speed (N), required thrust coefficient ($C_t$), and duct loading ($\tau$). It uses a seed file for blade geometry, which includes starting values for chord to diameter ratio (C/D), thickness to diameter ratio (T/D),
and two dimensional sectional drag coefficient ($C_d$) at several nondimensional radii. There is a similar seed file for the wake.

I’ve written a series of MATLAB scripts to generate PLL input files. This enables me to run a series of test cases to populate a given design space. This design space has dimensions of whatever inputs the user is able to vary, and the results of each test case can be displayed in the design space to help the user decide what values to use in his or her design. This is especially useful in applications that have few constraints, such as a thruster design in which the propeller and motor must both be chosen and matched, but neither has been decided on yet. The varied inputs in my current scripts are Z, D, U, and N. $\tau$ is held at 0.80. $C_t$ is calculated from the drag model of the vehicle in question. Contrary to intuition, $C_t$ does not vary with $U_{in}$. It can be found as a simple ratio between the frontal area of the vehicle ($A$), its drag coefficient, the number of propulsors the load is split between ($\Lambda$), and the swept area of the propulsor:

$$C_t = \frac{4AC_d}{\Lambda\pi D^2}$$

After the input files have been generated, I use a Windows scripting program called AutoIt (http://www.autoitscript.com) to execute each test case. My most recent script uses the response and log functionality of PLL to reduce the amount of “typing” the script has to do. This results in twice as many input files, but the filesizes are small and it prevents some hiccups that were present when the script input each value itself. It also allows me to capture a more complete output from PLL, including the number of iterations each test case takes to converge. This is especially useful in identifying non-convergent cases.

\[1\] this is true for bluff bodies, and mostly true for streamlined bodies, since the drag coefficient and frictional coefficient of a streamlined body are functions of velocity. The magnitude of the changes in drag need to be evaluated for each different vehicle model and determined to be negligible or not.

AutolPLL script running.

M Jordan Stanway • squall@mit.edu
Another MATLAB script parses the logfiles output by PLL and saves pertinent values to the workspace. The current version of this script reads the number of iterations before convergence (\(I\)), the total thrust generated (\(F_t\)), the propulsor efficiency (\(\eta\)), and the torque required (\(Q\)), for each test case.

\[ \eta_{\text{max}} = 0.575 \] @ \( U = 1.25 \text{ kts} \) 450 RPM

Output graphs for a representative test case: \(\eta\) vs. \(N\), \(\eta\) vs. \(J\), and \(Q\) vs \(N\).
Yet another MATLAB script plots the values for $\eta$ and $Q$ as functions of $N$ or the advance coefficient ($J=60U_{in}/ND$). Each combination of $D$ and $Z$ is graphed in a separate figure, with different lines for the different inflow speeds. $I$ and $F_t$ are used to filter out non-convergent solutions. On the figures showing $\eta$ vs $J$, the results for all inflow speeds overlap (as should be expected). The figures also report the maximum efficiency and the rotational speed and inflow speed at which it was reached, but this isn’t necessarily reliable, as some of the non-convergent solutions get through the filtering still.

The AutoIt script runs for 3-4 hours given a design space of about 15,000 test cases, and has a timer integrated into it. The input file generator and output parsing MATLAB scripts are fairly slow right now and I’m considering getting some help from a friend to rewrite them in Java or some other language. (I suspect the main reason they are slow is the way MATLAB has to deal with ASCII files.) That isn’t a high priority concern, though.

I’ve also run some testcases with different vehicle models, using basic drag calculations on the SeaBotix LBV and Hydroid Remus AUV to look at where we stand in relation to them. I plan to do the same thing for the Bluefin-9.

**Blade Design**

I’m looking through the results I have now to narrow in on a few combinations of $Z$, $D$, $U$, and $N$ to look at more closely. I’ll run a higher resolution design space on those combinations and investigate some changes in the duct design and also in spacing between the elements. After that, I’m planning on going through MTPLL to further validate 2-3 particular designs. Then, I will use PBD to finalize the blade design.

Once the blade parameters are fully specified, I’ll use another of my MATLAB scripts to export cross-sections in cartesian coordinates to SolidWorks. This is where I run into one of the biggest difficulties I’m having right now. SolidWorks will import the curves easily enough and loft a surface through them without any trouble, but for some reason it cannot loft a solid through them. The sad result is that I have hollow blades with no wall thickness. Not so much a problem in using them to make pretty pictures of the ROV, but when the propeller is exported to an .stl file (for 3D printing), it only keeps the hub. I am looking into several potential solutions for this. One is simply using another CAD program, Pro/Engineer, to make the propeller models. I’m learning the background of the program to try that right now. Another is to try and use design tables with 3D sketches in SolidWorks. This involves manually dimensioning 1000-plus points on the first template file, but would hopefully be completely importable afterward. I’m also investigating the mold designing package to see if I can just use the surfaces without having a positive solid to begin with, but this doesn’t look too promising. The last alternative
I've been able to come up with is generating an .stl or .igs or other comparable file directly from the MATLAB script. I'm not sure how feasible this is, but it would provide the best ultimate functionality, including portability to multiple CAD and modeling programs.

**Choice of Motor**

Thaddeus and I have been looking at many different manufacturers and finally found two motors that fit our expected requirements for size, power, and price. The motor from ThinGap is a little more expensive and has better power density, but it does not have factory solutions for gearing down, which will be necessary for our application. The Maxon motor, on the other hand, has a wide range of stock gearboxes available, and is a bit cheaper. The power density is not as high, but we don’t need it to be.

**RE 40** Ø40 mm, Graphite Brushes, 150 Watt

![SolidWorks surface loft and resulting hollow blade.](image)

Maxon RE40, 150 watt, 12 vdc brushed motor.
After confirming some things with the technical representative, Thaddeus has ordered one sample motor for us to work with. It is expected in the next week or two. He also has some worksheets of tested motor characteristics that he is going through to provide me with a more complete specification of its capabilities. I’ll use this to double-check the propeller design, but I’m using published threshold values on the motor/gearbox combination for now.

3D Printing/Mold Fabrication

I have been asking around MIT about 3D printers to make the molds for the propellers and have located two that I may be able to use. Dick Fenner has one in Pappalardo Labs that isn’t online right now, but should be partway through the term. I’ve just heard back from Col. Young in course XVI and I may be able to use theirs if we’re ready before the Pappalardo printer is online. Alternatively, we can outsource to one of the online fabrication companies (which I’m told have good turnaround and reasonable pricing) or possibly mill the molds out of metal stock. I’m going to schedule meetings with Fenner, Prof. Drela, and/or somebody at the Edgerton shop to talk about this possibility once I know more about what the props are going to look like. A quick look on McMaster and comparison to the cost for 3D printing powder indicates that both methods will be comparable in that respect.