CHALMERS

TOW Mode 2.0: Design of a speed controller for motor boats

Master of Science Thesis

ANDERS TWETMAN

Department of Signals and Systems
Division of Automatic control, automation and mechatronics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden, 2012
Report No. EX027/2012

TOW Mode 2.0: Design of a speed controller for motor boats

2012-04-20

Anders Twetman
EX027/2012
19840515-6756
Department of Systems and signals
Chalmers Universsity of Technology

Acknowledgemnts

I would like to acknowledge some people, without whom this project would not have been feasable. First I would like to thank CPAC Systems for giving me the opportunity to work on this project and above all my advisor Johan Oskarsson and Jonas Larsson who helped me with many practicle details rgarding implementation. I also wish to thank Lars Johansson who took over after Johan moved to stockholm. Furthermore I want to thank my examinor at Chalmers, Jonas Fredriksson, for giving me guidance when I needed it and of course also my family for supporting through this project and my years at Chalmers before that.

Abstract

The aim of this work has been to further develop TOW mode, an engine control program for power boats, used for towed water sports such as water skiing. The main development has been to introduce a constant speed regulator, to be used for wake boarding, in an already existing framework. This has been achieved by mathematical modeling and simulation of the boat along with controller design in Matlab/Simulink. The controller, along with a support structure in the form of a state machine and human machine interface, has been implemented on a preexisting hardware and software platform. Tests verify that the system works as intended and meets all requirements.

Contents

1 Introduction

Among users of small boats such as day cruisers and other recreational crafts, towed water sports i.e. wake boarding and water skiing are very popular. To get the most out of the sport, riders demand certain pulling characteristics of the boat. These vary between the different sports as water skiers want a smoother, wake free pull for slalom while wake boarders prefer a more aggressive pull that generates a bigger wake that they can use for jumping. To achieve these particular pulling characteristics the driver of the boat needs a certain amount of skill that few recreational boat drivers posses. Therefore there is a need for special functionality that can assist the driver in achieving this. Parts of such a function has already been implemented as a function called TOW mode, this work sets out to further develop TOW mode and implement the hitherto missing parts.

1.1 Problem definition

The existing TOW mode implements a simple limiter of the engine RPM which is suitable for the smooth pull needed for water skiing. What remains to be added is a mode more suitable for wake boarding that allows the rider to pull back on the boat which would then respond with a burst of power from the engine to keep the speed constant - an effect of this would be larger wake which is what wake boarders require. The problem can be defined as developing, implementing and testing a controller for regulating the boat speed to near constant. The controller must be integrated into an existing software environment and operate under a set of restrictions that are to be defined later.

1.2 Aim and goal

The aim of this work is to learn how to develop and implement a control algorithm within an existing framework of both hardware and software. The goal of this project is to implement a functioning speed controller that meets the requirements as stated in the requirements section.

1.3 Method

The theory needed as a basis for this project will be studied in the literature. Based on the theory a model for the entire system will be set up and a suitable controller will be developed. With the help of staff at CPAC System, the developers of the original TOW mode, the controller will then be implemented in the existing software system. Several exterior functions, such as a menu system will also be implemented that can be used for debugging and testing purposes. Through testing in a mock up rig the overall functionality can be verified. The functionality of the speed controller can be verified through simulation.

2 Specification

Based on the specification and requirements of the original TOW mode, a new set of requirements are stipulated. Before listing the requirements certain concepts have to be defined. There are two different modes of the TOW system, *rpm mode* wherein engine rpm is kept constant which is suitable for water skiers and *speed mode*, aimed at wake boarders with a speed regulator. *Activation* and *deactivation* refer to turning the *system* on or off while *engage* and *disengage* refer to turning the regulating function on or off. The *rpm limit* is the maximum allowed engine rpm when in rpm mode, the *speed limit* on the other hand is the set point used by the speed controller when in speed mode. *Nominal rpm* is defined as the engine rpm that corresponds to a certain speed under ideal conditions. The boat is controlled via a *Helm Control Unit* (HCU), which can either be amounted on the side of the helm, so called *side-mount* or on to, *top-mount*. Having defined these important terms, the system requirements are now listed.

2.1 Requirements

1. The system shall be available with the following prerequisites
   (a) There is a single engine with a single drive line
   (b) There is a single helm station - no station transfer allowed
   (c) The controlling HCU is either of Side-mount type or Top-mount with 2.5" info-display connected
   (d) A certain parameter is enabled in the data set for configuration, something only possible in a workshop
2. Activation of speed mode shall be possible when:
   (a) Prerequisites apply
   (b) Ignition is on
   (c) Lever is in neutral position
   (d) Rpm mode is not active
3. Activation of rpm mode shall be possible when:
   (a) Prerequisites apply
   (b) Ignition is on
   (c) Lever is neutral
   (d) Speed mode is not active
4. Deactivation shall be possible when:

(a) System is active

(b) Lever is in neutral position

5. The system shall engage when:

   (a) The system is active

   (b) Engine is in forward gear

   (c) The lever position is such that the rpm would be equal to or larger than the limit for the chosen mode - that is, rpm limit or nominal rpm - if the system was inactive

6. The system shall disengage when:

   (a) System is active and engaged

   (b) Lever position is such that rpm falls below the limit rpm of the chosen mode

7. When TOW mode is engaged, the following must apply:

   (a) The boat speed shall be regulated to the set point if the system is in speed mode.

   (b) The engine rpm shall be set to the value of the rpm limit if the system is in rpm mode

   (c) If the set point / limit is changed, the engine rpm shall immediately be changed accordingly

   (d) If the rpm limit / nominal rpm is raised above the rpm corresponding to current throttle position, the system shall disengage.

8. The following genera requirements must apply at all times:

   (a) Only one mode can be active at any time, either speed mode or rpm mode

   (b) Only the active mode can be engaged

   (c) When the system is disengaged, the engine shall respond to user command as if the system were inactive

   (d) As long as the system is engaged, the position of the throttle lever shall not be taken into consideration other than for disengaging

   (e) The system shall only engage when the engine is in forward gear, in reverse the throttle shall respond to driver command as normal.

   (f) The user shall only be able to change set point or rpm limit when the system is active.

2.2 State machine

Using the above requirements, with the previous version of TOW mode as a template, a finite state machine representing the overall structure of the controller is designed. As can be seen in figure 1, there are two paths depending on which mode is chosen, rpm or speed. Each mode also has a hierarchical structure within the active state, such that the system can be engaged or disengaged while being active at the same time.

The finite state machine only shows the states and the transitions, to get a more fleshed out structure, actions must be added to the states. Figure 2 shows a simplified state machine - optional path depending on mode and hierarchical states have been removed - with actions added to the states. The same thing has been done for the inner state machine in figure 3.

As seen from these three state machines, if all prerequisites are fulfilled, TOW mode is available and the system starts as inactive. If the lever is in neutral and the user request that TOW mode be switched on, the system activation in progress. Depending on which mode is requested there are two possible activation paths, one for speed mode and one for rpm mode. Activation in progress takes care of all *Human Machine Interface* (HMI) actions, such as sounding the buzzer, turning on the TOW LED on the side-mount panel and indicating that TOW mode is on, in the menu screen. From here, the systems moves directly to the active state where the controller listens for user commands and acts accordingly e.g. changing the set-point. If the user requests that TOW mode is turned off and the lever is in neutral, the state is changed to deactivation in progress which takes care of the HMI such as turning off the TOW LED, then immediately changes the state to inactive. Some other functions, such as cruise control, are allowed to interrupt TOW mode, if this happens there is a state change from active to abandoned and from abandoned to deactivation in progress.

Except for checking user input, the active state also runs the inner state machine which starts in the disengaged state. When the user brings the lever forward, past the limit point the system engages and either regulates the speed or sets the rpm limit, depending on the mode. As soon as the lever is brought back behind the limit point, the system switches back to disengaged. Since this state machine is not directly connected to the outer state machine, the user is able to send commands to the system regardless of which state the inner state machine is in.

2.3 Human machine interface

The Human Machine Interface needs to be changed in order to incorporate the new functionality of TOW mode. The main changes will take place in the menu system. Before any changes of the menu system are implemented, a general understanding of said menu system is needed. A generic menu screen is shown

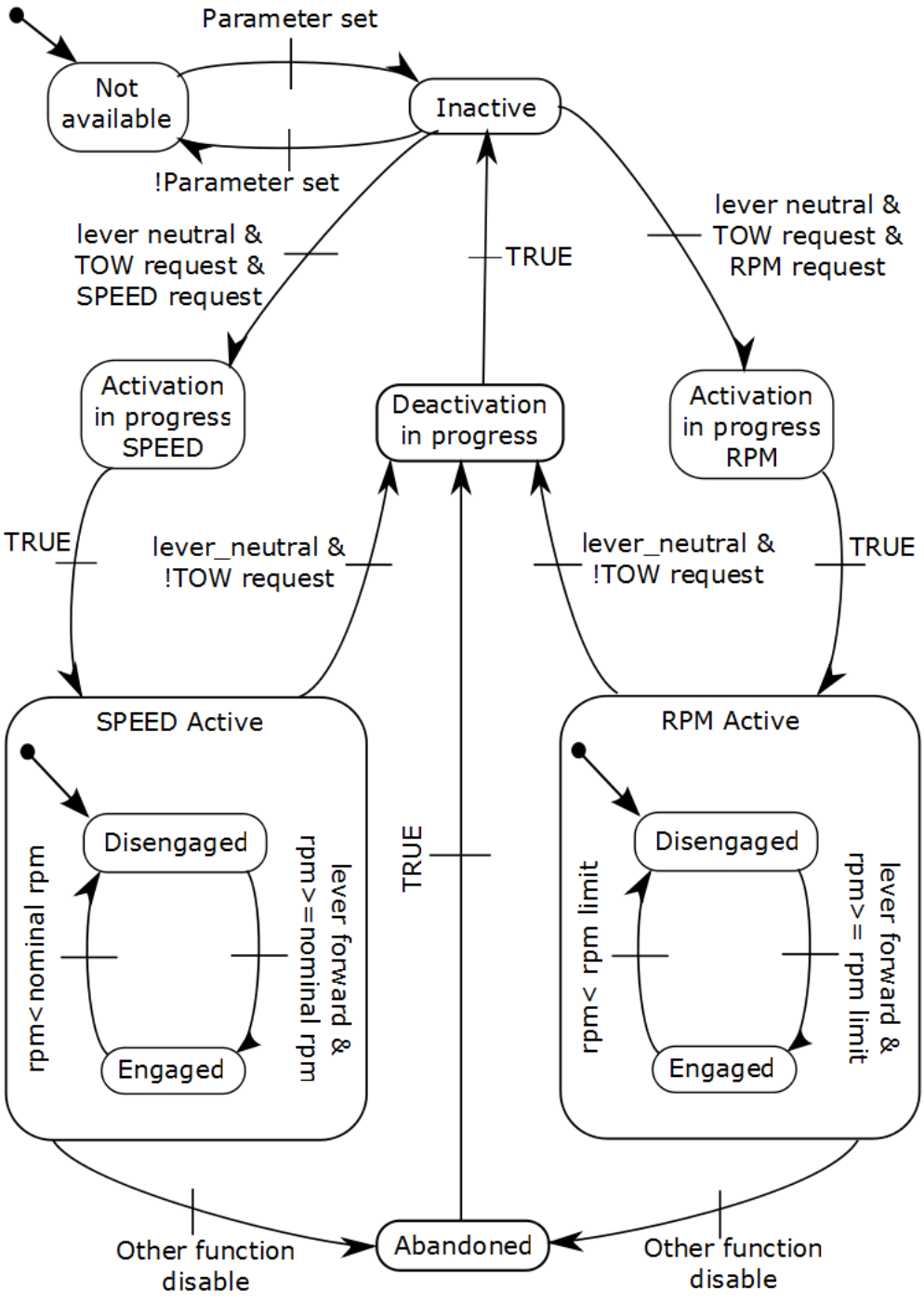

Figure 1: *The finite state machine*

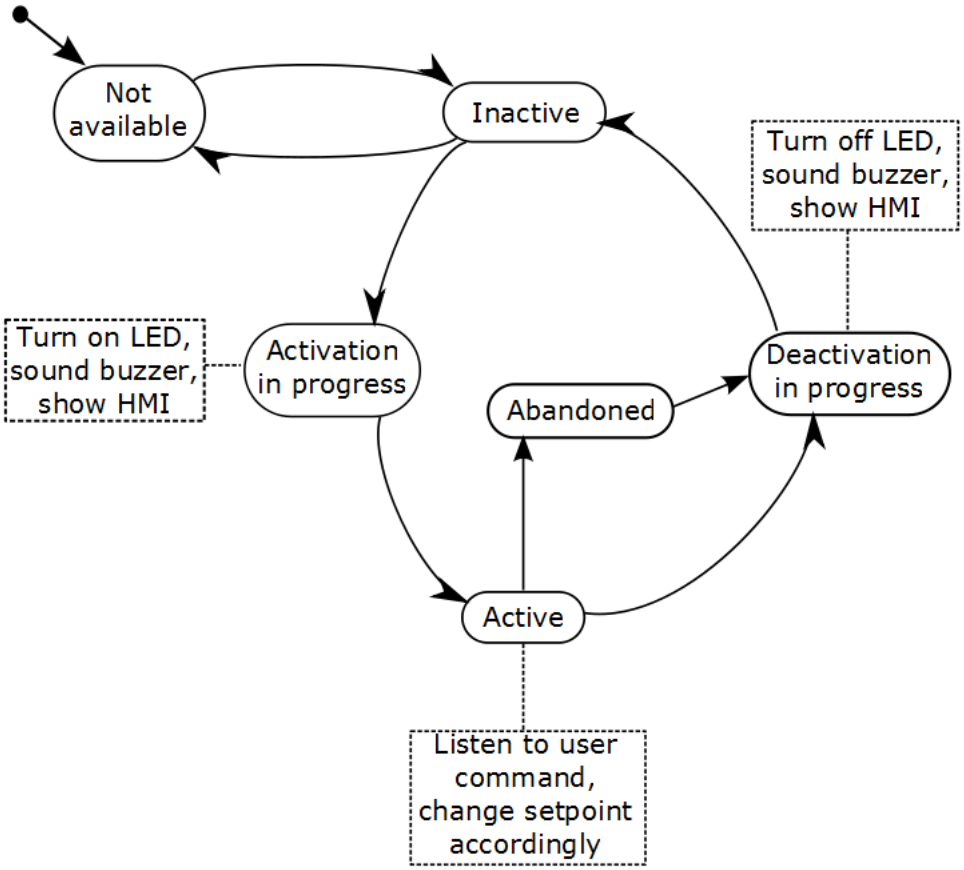

Figure 2: *Simplified state machine with actions added*

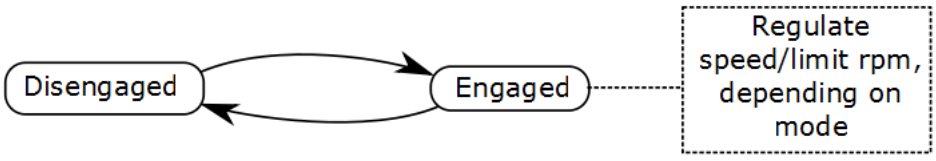

Figure 3: *Simplified state machine for inner loop, with actions*

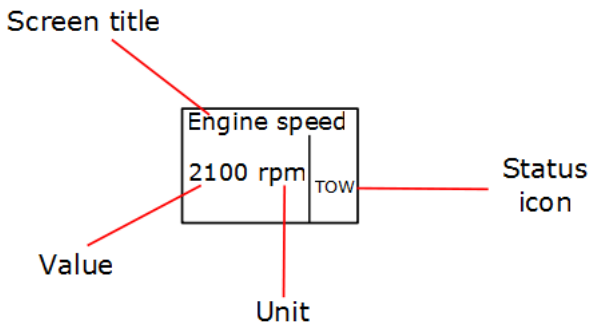

Figure 4: *Generic menu screen*

in figure 4, several similar screens can be navigated by four push buttons, BACK, LEFT, RIGHT and OK. LEFT and RIGHT are used for switching screens, OK is for turning functions on or off, as well as entering sub menus. BACK is used for exiting sub menus. For TOW mode there are also special buttons on the side-mount HCU, PLUS and MINUS for increasing or decreasing the set-point and TOW for switching the system on and off.

When TOW mode is inactive, the TOW menu screen should be blank except for the title and any status icons from other functions. Pressing the OK while lead to a sub menu with two screens, one for speed and one for rpm. Which one of them is shown depends on a flag for mode default, if turned on, default is speed mode, otherwise rpm mode. A second push of the OK button confirms that it is the selected mode that the user is interested in and brings up an indication that the system is currently turned off. A final press of the OK button turns the system on with accompanying LED light and buzzer as specified earlier (section 2.2) but also a pop-up screen with the text "TOW mode activated". Turning on TOW mode with use of the side-mount button brings forth the same pop up, then switches the menu screen to show this one, regardless of what screen was shown before the button was pushed.

Once the system has been activated the LEFT and RIGHT buttons, on the display, can be used to increment or decrement the set point. The PLUS and MINUS buttons on the side-mount HCU serve the same function but do not require that the user enter the sub menu first. When TOW mode is turned on, the BACK button skips the sub menu for choosing rpm or speed mode and jumps directly to the main menu, likewise the OK button goes directly from the main menu to the lowest sub menu. While TOW mode is active, the TOW screen in the main menu shows the set point as well as the unit i.e. knots or rpm depending on mode. Additionally, all menu screens should show the TOW status icon while TOW mode is active. A last addition to the menu system is an option in the settings menu, allowing the user to change the default flag from speed mode to rpm mode. In figure 5 there is a sketch of the TOW menu tree. The symbol

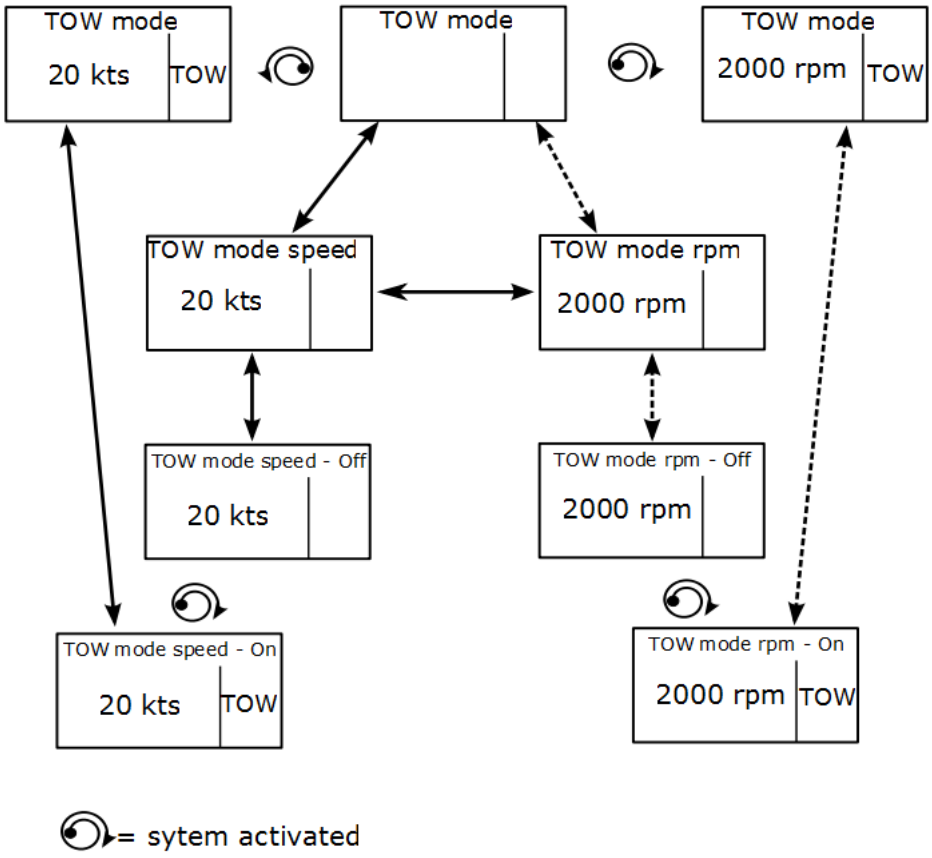

Figure 5: *Sketch of the TOW menu tree*

for system activated shows the difference between the inactive and the active state. The whole lines show default behavior while a dashed line represents the alternative path. This menu tree is weaved into the existing menu system.

3 Theory

The objective of this paper is to construct a speed controller for a small motor boat, to do this some background knowledge of boat propulsion and speed calculations is needed. This follows below.

The basic principal behind boat propulsion is very simple, the spinning propeller accelerates a column of water from the the fore of the boat and shoots it out behind. [Phillips-Birt]By Newtons law of reaction this then imparts a force along

the drive shaft of the propeller called thrust and since the drive shaft is connected to the engine which in turn is connected to the hull, the thrust pushes the boat forward. The details of this process involve some rather complex fluid dynamic calculations but for the purpose of this paper a brief overview is sufficient. In fact, the mathematics are so complex, and in some cases incomplete, that most naval architects and boat builders alike rely heavily on model test and well documented standard series.

3.1 Attainable speed

What speed a given boat will reach depends largely on size i.e. length and displacement, the hull shape and fullness of the hull (given by the block coefficient), center of gravity and of course the propulsion unit, that is, the engine and propeller [Tornblad]. The most essential of these factors is the power to weight ratio, kg/kW or LB/HP, simply put, more power per unit weight gives more speed. When considering power one must note that there are several losses in the transmission, lowering the effective power at the propeller. Also note that continually running an engine at top speed is ill-advised, therefore the engine power should be dimensioned such that cruising speed of the boat can be maintained at $80 - 90$ % of engine top speed; this also gives a power reserve for dealing with unexpected conditions such as harsh weather. Closely connected to the power to weight ratio is the so called speed length ratio: $\frac{V}{\sqrt{WL}}$, V is boat speed and WL is the length at the waterline. The speed length ratio (SL ratio for short) shows that it is easier to achieve high speeds if a craft is long and thin rather than short and thick. For non-planing vessels, reaching a speed length ration above 1.4 is almost impossible no matter how much power is installed, largely due to the fact that getting more power means using a bigger, heavier engine thus slowing down the vessel [Gerr].

The word non-planing refers to boats of displacement or semi-displacement type. A displacement boat is one where the entire weight of the craft is held up by the hydrostatic buoyancy, all according to Archimedes famous principle. If a craft has a low displacement for its length - a low displacement length ratio - and a suitable hull shape it can reach speed length ratios of around 2.5 or even 3.0, this classifies it as a semi-displacement craft [Gerr]. Speed length ratios of 3.0 or more are only attainable by true planing craft, such vessels have a completely flat under body aft, high power and light weight. The result is that, at high speeds the boat is held up, not by the buoyancy but by the hydrodynamic lift – it planes, hence the name. Most modern powerboats and leisure craft of the type relevant for this paper are planing craft. The speeds this type of boat can achieve can be calculated by Crouch's Planing speed formula, see Gerr for more details.

3.2 Propeller

3.2.1 Diameter and rpm

So, a given hull can be driven at a certain speed depending on its speed length and displacement length ratios and engine power. However, this only specifies the attainable top speed, to calculate the speed at any given moment one must also consider the propeller. According to Gerr "the single most critical factor in determining the amount of power that a propeller absorbs and transmits" is the diameter. The amount of power absorbed by the propeller in turn determines how much thrust it delivers to the boat and thus the boat speed. Under normal circumstances a larger diameter yields a higher efficiency, for high speed vessels however, the increased drag of a large propeller might decrease the efficiency.

The larger diameter the propeller has, the larger the amount of torque needed to turn it becomes, as the torque increases, the rotational speed decreases so that a large propeller will turn very slowly. For high speed craft such as the ones considered in this paper, using a small propeller and turning it at high speeds is often beneficial considering drag. This also eliminates the need for a reduction gear which would normally reduce engine rpm (revolutions per minute) to suit the propeller size; in doing so some power is wasted.

3.2.2 Pitch

Another factor governing propeller thrust, and consequently boat speed, is the blade pitch. A popular analogy for a propeller is that it acts like a screw[1] being screwed into a soft material e.g. a wood screw in soft pine. During one revolution, it will then drive itself forward a certain distance, this distance is the pitch. Since the propeller is attached to the shaft which in turn is connected to the hull via the engine or shaft bearings, the boat is pushed forward the same distance as the pitch for every revolution [Gerr]. This gives a theoretical speed of $V = P * N$, V is speed, P is pitch and N is shaft velocity, but it is important to point out that this speed is purely theoretical as shall be explained later.

The pitch can also be described as the angle of incidence between the hydrofoil (the propeller blade) and the water stream. The hydrofoil acts much like an airplane wing, creating low pressure on one side and high pressure on the other this accelerates the water from the low pressure side (ahead to the high pressure side (astern). With increased angle of incidence (pitch) comes increased suction and pressure[Phillips-Birt]. Thus the pitch converts the shaft torque to thrust by deflecting or accelerating the water with higher pitch yielding more thrust [Gerr]. This works by newtons second law, thrust is equal to mass of the water times the acceleration.

Other factors such as the number of blades, the area of each blade, the shape of the hydrofoil section and so on all affect the propellers performance in different

[1] The proper name for a boat propeller is actually screw.

ways, however for the kind of leisure craft that this paper is concerned with, the propellers used are almost always standard three bladed ones; in essence pitch and diameter are the only varying factors.

3.2.3 Slip

The analogy of a propeller working as a wood screw is a vary noble way of describing it, however, water, unlike wood is fluid and the propeller will slip a little when it turns. This means that the true speed of the boat is slightly less than the theoretical $V = P * N$. The slip is normally expressed as a percentage of theoretical speed and can be as high as forty five percent for low speeds while it decreases exponentially for higher speeds until a value of around ten percent. How slip is calculated varies between different methods. Crouch's propeller method, which is the oldest and most common, is based around calculating the apparent slip, simply the theoretical speed minus actual boat speed. The pitch of the propeller is then adjusted to compensate for this slip [Gerr].

The Bp-δ method, while not as widely spread as Crouch's method nor as well proven, is considered more accurate. This method takes consideration of the fact that water sticks somewhat to the hull due to friction and is dragged along a bit, creating the wake. This means that the propeller advance is slightly less than boat speed, giving the real slip as apparent slip plus the wake. As mentioned earlier the fullness of the hull is one of the factors determining how fast a boat will go. The fullness is described by the block-coefficient which can be used to calculate the wake. Through a set of calculations and with the help of Bp-δ diagrams a suitable propeller, which compensates for both slip and wake, can be chosen[Gerr].

3.3 Factors determining speed

Thus, given certain hull characteristics, engine and a correctly chosen propeller, a boat will be able to cruise at a previously specified speed while maintaining a suitable power reserve. Assuming that all of these factors have been adequately chosen by the boat builder for the particular boat in question, we can now concentrate on controlling the speed. For that we need to know the relationship between speed and drag and, since the controller works with the engine rpm as control signal, the relationship between rpm and speed.

Taking into account that the propeller has been chosen to compensate for the effects of slip and wake, simply assume that boat speed is proportional to shaft rpm which, without any reduction gear is the same as engine rpm. The proportionality constant is then the adjusted pitch $V = P_a * N$. To calculate the drag of the boat is considerably harder. Drag, or resistance, consists of three components, the wind resistance of the part of the boat above water, the friction resistance against water and the wave making resistance. Interesting to note

is that total resistance varies greatly with the speed length ratio, taking into account that most planing craft are made to cruise at high speed length ratios (that is the definition of a planing craft) one naturally assumes that resistance will be fairly large [Phillips-Birt].

The frictional resistance is comparatively large at lower speeds, as much as ninety percent of the total but decreases rapidly with higher speed length ratio, for high speeds the wave making resistance dominates completely. The frictional resistance depends, in essence, on area of the hull, the length of the vessel, the type of surface and of course speed. Even a hull that is made smooth to start with will get slightly foul over time due to build of algae, sediment and the like on the hulls surface. For this reason the friction increase more or less quadratically with speed, although a very smooth surface might get a slightly lower exponent [Phillips-Birt].

As a vessel moves in water change in the pressure occur that form waves, part of the engine power will then be spent overcoming the pressure forming these waves, this is what is known as the wave making resistance. The wave pattern differs for various speeds but at any given speed the pattern will always be the same. The wave system consists of both divergent waves that move away from the stern and bow at an angle, and transverse waves that are parallel to the line of travel. It is the transverse waves that stand for the most part of wave resistance. The speed of the waves is related to the wavelength by $V_w = 1.34 * \sqrt{\lambda}$ V_w is wave speed, $\lambda$is wavelength. This means that, at a speed length ratio of 1.34 the boat will be supported by one crest at the bow and one at the stern, it rides one wavelength. At higher speeds it leaves one wave behind and the stern will dip into the hollow, this causes the power needed to overcome the waves to rise rapidly[Phillips-Birt]. Hence the wave resistance increases exponentially with boat speed.

So far the discussion has concerned friction resistance and wave resistance only but according to Phillips-Birt wind resistance is not inconsequential, at high speeds it can be around ten percent of the total. Now, this is not a great amount compared to the other two but it should not be forgotten. Minimizing top hamper in the way of cabins, fly bridges etc. will of course lower wind resistance but it can not be done away with altogether. In an example given by Phillips-Birt, the wind resistance quadruples with a doubling of wind speed so it would be safe to say that it increases roughly quadratically with boat speed. From this we can deduce that, even though the exact relationship between speed and resistance is not know, all the components seem to vary exponentially with speed. Therefore, let us state that resistance, or rather drag, is given by $D_f = bv^2$, D_f is drag force, b is proportionality constant and v is boat speed. That this assumption holds true is evident from any one of the speed-power curves found in for example Gerr or Phillips-Birt. Theses curves show the amount of power needed to achieve a certain speed given that all other factors remain the same, it is evident from these curves that drag increases exponentially with speed.

4 Modeling, simulation and controller design

The necessary background knowledge for a speed controller has been given above, the next step is to use that knowledge in creating a mathematical model of the boat system. As mentioned earlier (section §3) the process of driving a boat using a propeller is very complex, so far no accurate mathematical models exist and most naval architects use scale model testing to a large extent. Therefore setting up the complete model of the propeller, the engine and the hull would be very time consuming and demand a lot of work. One could of course use the model set up by Browning which takes many factors into consideration but for this approach to work one must have access to a large number of parameters. In this project, the model has to be valid for a large range of boats, most of them unknown, which makes such a detailed model impractical. A simpler model of the boat as a whole is used instead. A general model of a boat is based on six degrees of freedom as done by [Browning] but for the purpose of a speed controller three degrees of freedom is more than enough, in fact [Xi & Sun] claims that surge motion can be decoupled from heave and pitch thus only necessitating one degree of freedom. However, the model used by Xi & Sun does not consider surge motion, so constructing a model from scratch becomes necessary.

4.1 Mathematical model

Going back to basic control theory and the simple methods of modeling found in e.g. Lennartsons text book one finds that, for simple systems, a balance of forces equation, wherein the change in momentum per time unit is calculated as the difference between driving forces and resisting forces, will yield a sufficiently accurate model. That is: $\begin{bmatrix} Change\,in \\ momentum \\ per\,unit\,time \end{bmatrix} = \begin{bmatrix} Driving \\ forces \end{bmatrix} - \begin{bmatrix} Resisting \\ forces \end{bmatrix}$

Using the sketch in figure 6 we find the forces acting on the boat and can set up the following balance equation:

$$m\dot{v} = T\cos(\tau + \varepsilon) - D_f \cos\tau - N\sin\tau$$

In most situations the boat's trim angle $\tau$is relatively small, around 3 degrees, and the same thing applies to the propeller shaft angle ε. For small angles, $\sin \approx 0$ and $\cos \approx 1$ so the above equation can be simplified to

$$m\dot{v} = T - D_f$$

From section 3.3 we know that resistance varies quadratically with speed i.e. $D_f = bv^2$so, inserting this into our last equation we arrive at $m\dot{v} = T - bv^2$which can be rearranged to

$$m\dot{v} + bv^2 = T$$

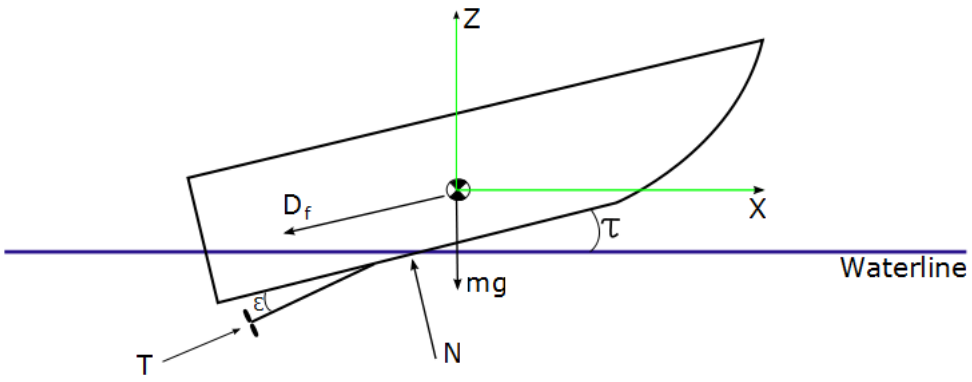

Figure 6: *Sketch of the boat system. Coordinate axes X and Z, trim angle τ and propeller shaft angle ε. Forces: thrust T, water pressure N, drag D_f and gravity mg.*

Assuming that this equation is valid in the time domain, the proper form would be $m\dot{v}(t) + bv^2(t) = T(t)$. Most controllers are based around linear systems, this one however is nonlinear, which poses a problem. In this case it is easily solved by linearizing the system around a point; this is possible since we are interested in regulation rather than servo following, that is the reference value is expected to stay constant. So, linearization around the point v_0, $\dot{v}(t) = 0$ for $t = 0$ and $T_0 = bv_0^2$ yields the new system $m\Delta\dot{v}(t) + 2bv_0\Delta\dot{v}(t) = \Delta T(t)$. For practical purposes linearization point $v_0$can be set to 1 since the output will be scaled later on. The equation is then Laplace transformed, as the Laplace domain better lends itself to control problems. Using some sloppy notation to remove the Δsymbols, we now get.

$$msV(s) + 2bV(s) = T(s) \iff (ms + 2b)V(s) = T(s)$$

Using propeller thrust T as the input and boat speed V as the output we now generate the system transfer function

$$\frac{V(s)}{T(s)} = \frac{1}{ms + 2b} = \frac{\frac{1}{2b}}{\frac{m}{2b}s + 1}$$

This is more or less a standard first order system which is not surprising considering the nature of a powerboat, incidentally it is very similar to the one used to represent a car in Lennartsons example. Note however, that the input to this system is the propeller thrust T, the available hardware on the other hand does not allow us to dictate thrust, only engine rpm. Thus we need a function relating

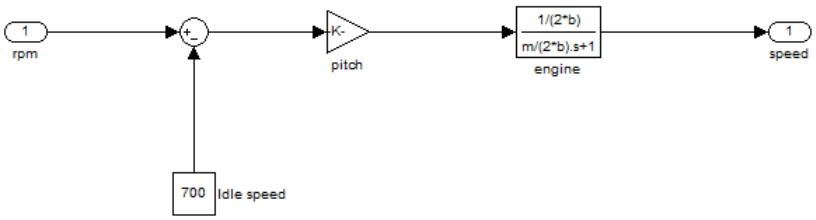

Figure 7: *The Simulink model of the boat*

rpm to thrust, finding one is easier said than done, what we do have (section 3.3) is a function relating boat speed to propeller pitch, $V = P_a * N$. Assuming that the propeller, motor, gearbox and so on all have been correctly chosen to deliver the amount of thrust needed to drive the boat at a certain speed this simple relationship should suffice. Before incorporating it in the system model we must compensate or the engines idle speed. When the engine is idling it will be turning at a very low rpm without moving the boat. Supplementing the equation with a constant $N_0$for idle speed we get $V = P_a * (N - N_0)$.

We now superimpose the conversion from rpm to speed on our first order system to generate the complete model of the boat.

$$V(s) = \frac{\frac{1}{2b}}{\frac{m}{2b}s + 1} * P_a(N - N_0)$$

4.2 Simulation and controller design

Using the mathematical model, the system is now simulated to verify the overall behavior; for this purpose, Matlab and Simulink are used. The basic boat model can be seen in figure 7, when simulated with rpm as input it gives the characteristic output of a first order system, somewhat modified by the rpm to speed conversion, see figure 8.

A disturbance (it could be a current, wind or a wake boarder), in the form of white noise filtered through a very simple first order system, is added along with a simple set-point to rpm conversion (reverse of $V = P_a * (N - N_0)$) acting as an open loop controller, see figure 9details. Simulation of this system yields a more typical open loop behavior as shown in figure 10.

Having done some initial simulation, it is now time to design an appropriate controller. For a simple first order system like this, a PID type controller should suffice. Because of the conversion from rpm to speed, conventional design methods will not work properly, a simpler trial and error method is chosen instead. Due to the nature of the system, a proportional controller will do the job, but better results can be achieved by adding integrator action. The previously mentioned set-point to rpm converter is used a feed forward controller that provides

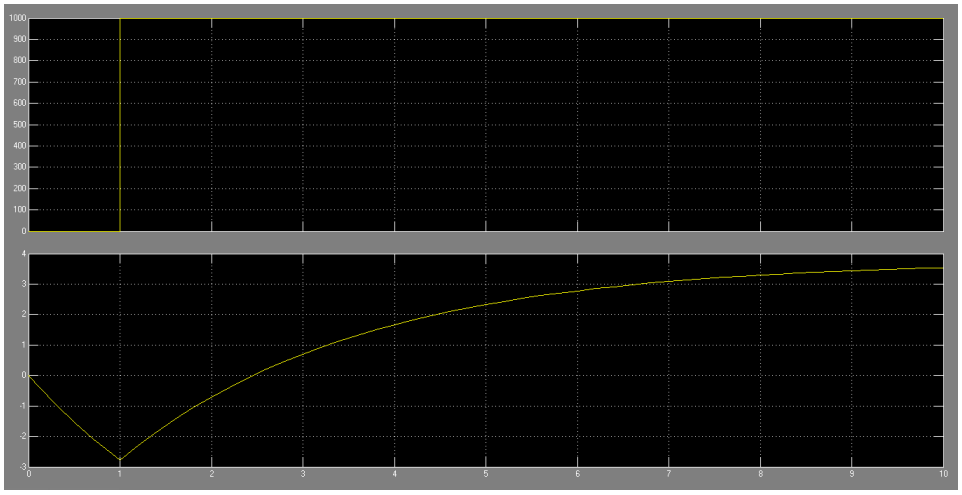

Figure 8: *Unregulated simulation output*

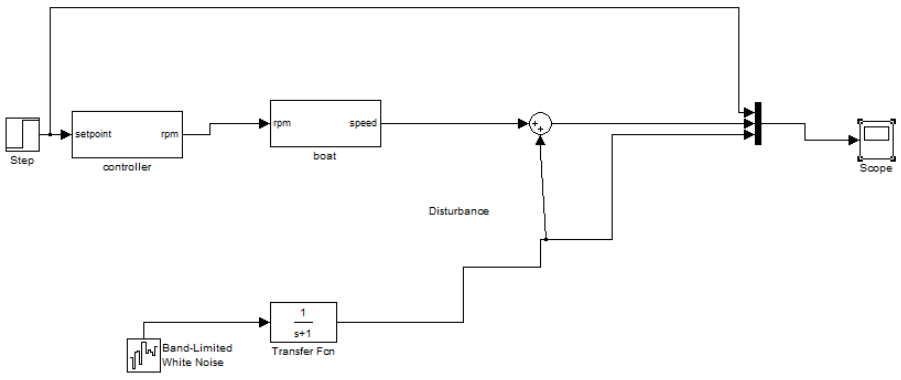

Figure 9: *Open loop system with disturbance*

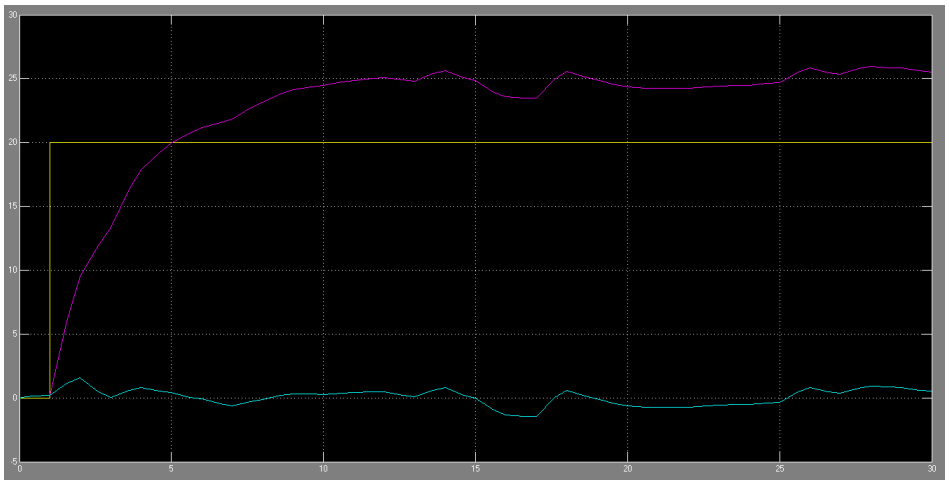

Figure 10: *Open loop simulation with disturbance*

a baseline for regulation; the proposed PI controller is then superimposed on this baseline (see figure 11) then tuned by trial and error using simulation results. To complete the system, the same filtered white noise as in figure 9 is added as a disturbance, yielding the system in figure 12.

This system is now simulated to further tune the controller with the aim of repressing disturbances while keeping control output low. The final simulation results are seen in figure 13.

Looking at figure 13 it is easy to see that the controller is working well, despite some rather large disturbances, the boat speed stays within ± 1.5 units of the set point, which is acceptable. Further simulation shows that this remains valid independent of the set-point, even when the disturbance to signal ratio grows (see figure 14). Note also that even though the control signal varies greatly, it never becomes saturated. Such a controller would have a satisfactory performance.

So far the design has been concerned with a continuous system, however, since the controller will be implemented in a computer, care must be taken to make sure that it will work as intended when it is transformed to discrete space. The main problem here occurs in the conversions between continuous and discrete space. The signals are measured (sampled) at certain intervals and these samples give an instantaneous image of what is going on, what happens between samples is unknown. It is simple to see that taking too few samples might lead to a lack of information about a signal. There might, for example, be hidden frequencies that cause a seemingly stable system to become oscillating when sampled. Converting the output from discrete to continuous might also cause problems. In most cases, this is done via a so called zero order hold, it simply holds the output value constant over one sample period. If anything happens

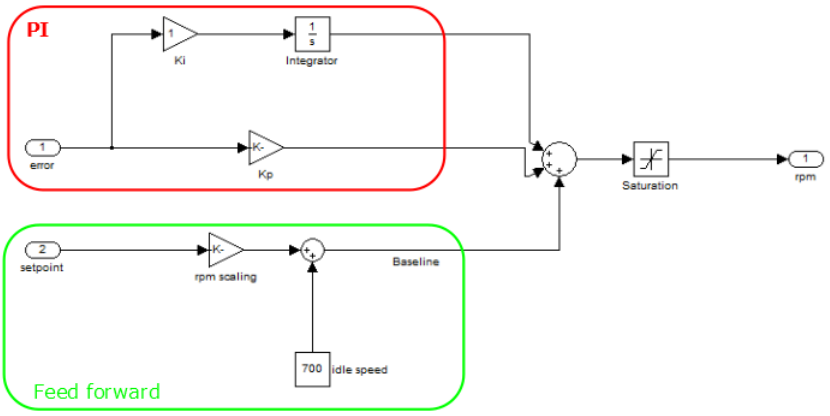

Figure 11: *The controller in Simulink*

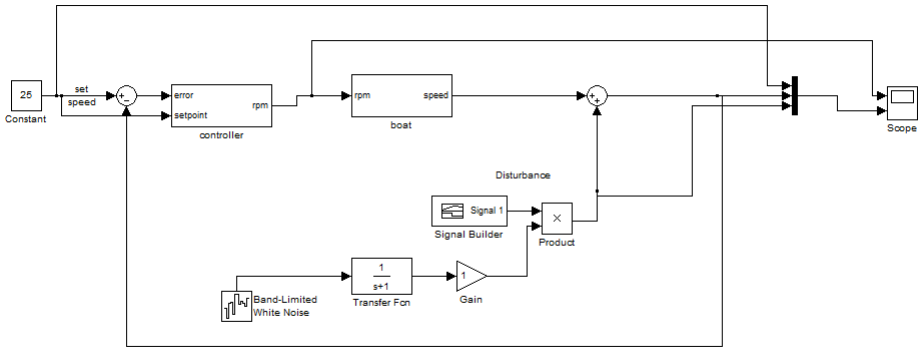

Figure 12: *The complete Simulink system, with disturbance.*

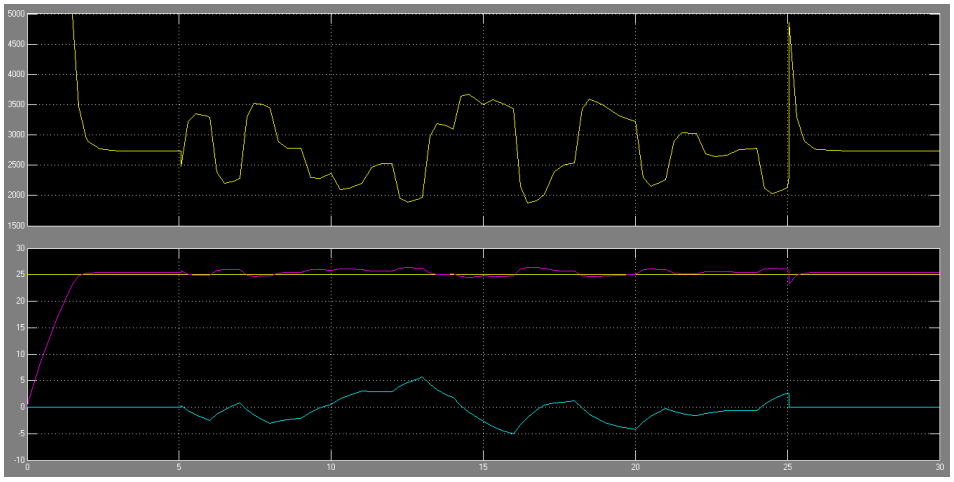

Figure 13: *Simulation output with disturbance and controller*

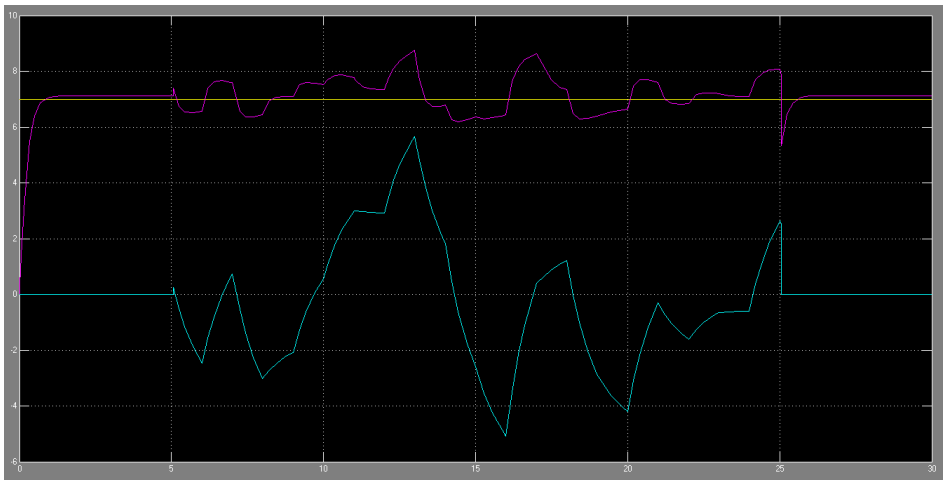

Figure 14: *Simulation with higher disturbance to signal ratio.*

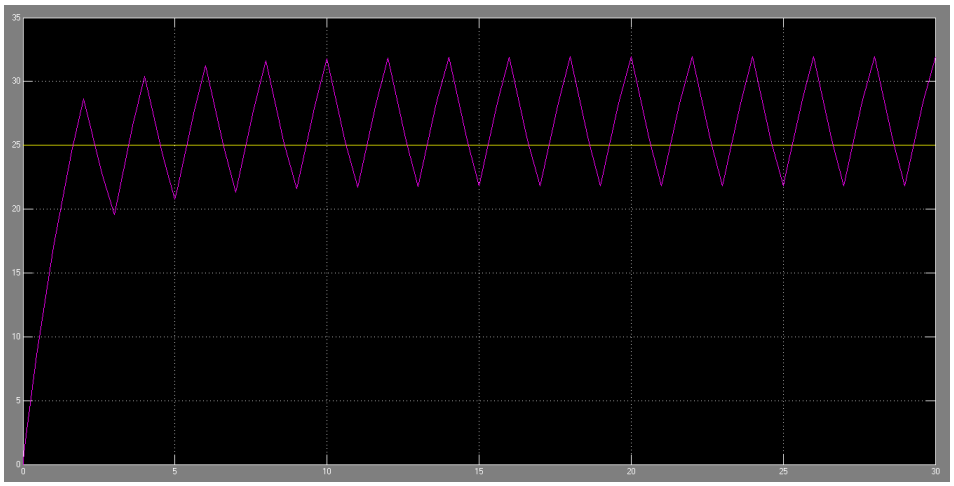

Figure 15: *Example of too low sampling frequency, no disturbance*

in between two samples, the system has no possibility to react.

A very severe problem which is important to avoid is that of Aliasing which occurs if the sampling frequency is too low. What happens is that signals of different frequency have the same values at several sample instants. When the signal is transformed back to the continuous domain, it might not be the same as the signal going in; the samples are not unambiguous. Examples of this and other problems that may occur due to sampling can be found in any textbook on digital control theory for example [Åström & Wittenmark]. The solution is to carefully choose a correct sampling frequency. In this project however, the choice of sampling frequency is limited by the overall software system, see section 5 for further details. The task then changes from determining a correct sampling frequency to testing the validity of the preselected frequency.

By changing the system in figure 12 to include a sampler on the sensor side and a zero order hold between the controller and the system it is possible to test the effects of different sampling frequencies. Using default sampling frequency of one hertz, yields the output in figure 15, clearly showing what too low sample rate can do to a system. Simulation with the proper preselected sample time of 16 ms results in the output in figure 16. Comparing with figure 13 one can see that the system output is nearly identical with the continuous case, the control signal however, becomes more oscillatory in nature. This is something that could be of concern when considering wear on the engine, but for the purpose of the control system, using the preselected sample rate is not a problem.

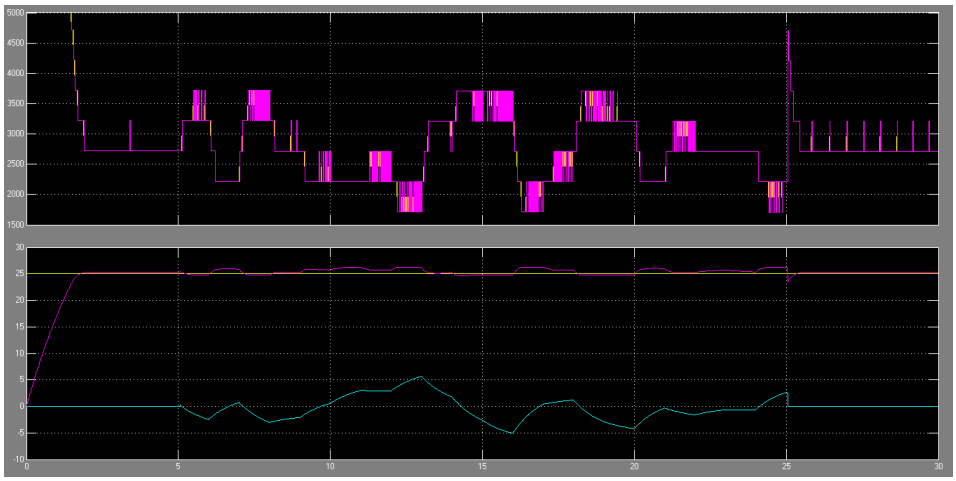

Figure 16: *System output with 16 ms sample time*

5 Implementation

A very large part of the project is the actual implementation of the controller in the existing platform provided by CPAC Systems. Certain aspects as well as problems, of the implementation process are discussed in this section . The hardware platform consists of a helm control unit, a *power control unit* (PCU), a 2.5" display panel, an ignition key and one or more sensors for speed measurement. These units are all connected via a so called CAN bus. CAN is a type of communications network widely used in the automotive industry for controlling such things as the air condition or the alarm and lock system. Describing the details of the CAN system is not part of this report, suffice it to say that all sensors, control units, displays and so on are nodes in a network with only few wires, and that commands and data are sent as clearly defined messages. The HCU holds the combined gear and throttle lever as well as some push buttons, it is also here that all the main functionality lies. The PCU is responsible for controlling the engine and communicating with sensors, it takes commands from the HCU. The menu system is a separate entity built into the display panel and works more or less independently from the HCU but data can be sent between the two. Finally the sensors are connected to the CAN network and data is formatted to fit the network messages.

Since the controller is implemented within the system provided by CPAC Systems, several functions already exist such as sensor input, motor output and human machine interface. With so many functions already provided, implementations is greatly simplified as focus can be laid on the main task at hand, not having to develop all the auxiliary functions needed. However, it also poses a barrier in the form of vast amounts of unfamiliar source code that has to be studied and analyzed before utilization is possible. This of course takes time.

Some auxiliary functions do not exist however, and some do not work fully in the way necessary for this project so new ones have to be constructed, the process of doing so is greatly simplified since existing functions can often be used as examples or templates. The most important functions provided are the input from sensors - they can be read directly as scaled and formatted values - and engine output via the PCU, for example a certain rpm can be requested from the engine without further thought on how to achieve this.

As the different hardware units communicate through the CAN network, there is no longer a need to be concerned with the specifics of each sensor/actuator or how I/O operations work at the lowest level, one simply transmits a properly formatted CAN message on the bus and the correct unit will pick it up and process it accordingly. The downside of such a system is creating proper message frames and using them to send or receive data. Luckily a large variety of message frames as well as functions for transmitting and receiving data are already part of the system. Therefor there is no need to deal with the CAN network as such for implementing TOW mode.

One aspect of great concern when implementing a controller in a computer system of some sort is the real time environment. As discussed in section 4.2 a slight change in sample time for the digital controller can have some drastic effects on system performance, so it is imperative that the controller functions is run at proper intervals to achieve this sample time. It is also important that controller is not interrupted for too long while running by e.g. a preempting function with higher priority. Note though that the TOW controller in particular is not critical as it is a leisure function; it can be defined as having soft real time criteria i.e. it is better that control signals arrive late than not at all. This gives quite a lot of leeway when it comes to timing and priority.

In the CPAC Systems software framework, the real time aspects are of course provided for already. The way it works is that there are several tasks scheduled to run at certain intervals, namely 8 ms, 16 ms, 32 ms and 256 ms. Furthermore the tasks within one interval are subdivided into four different priorities from highest to lowest. When to run any given function is determined heuristically. TOW mode has been preselected to run in one of the 16 ms tasks, and as seen in section 4.2 this sample rate is satisfactory for the controller. There are other 16 ms tasks with higher priority than TOW mode so it could be preempted, but as mentioned earlier, it is not a critical system so this should be acceptable. When the 16 ms task runs TOW mode, it starts the entire system, not just the speed regulator so there might be other parts of the system, such as checking user input, increases the execution time of the task. This is not a problem however, simulation shows that the regulator works, even with slightly longer sample intervals.

The hardware in the PCU does not support the use of floating point numbers, everything is done with integers. Some of the mathematical operations used in TOW mode demand floating point, however. There are special techniques for circumventing this problem in software such as scaling the values up, doing the

operation then scaling down again. Most of the situations where floating point numbers are needed have been resolved, however, one major area remains. The reading of the boat speed from the sensors is done in steps of one knot, which affects the set point error, effectively this causes the regulator to respond only when the boat speed differs from the set point by one knot or more. However, this is not a problem, essentially the lack of floating point numbers in the calculation of the error acts as a hysteresis, filtering out noise and small disturbances that might otherwise cause the system to start oscillating.

One more thing to note about implementation is the need for a speed simulator to be used for testing. As development work is done in an office without a real boat at hand, there is neither an engine or real sensors. The engine has been replaced by a simulation unit that responds to PCU commands in proper way such that rpm for example changes the way it would on a real boat. The sensors on the other hand are represented by variable resistors, giving out a constant value depending only on the position of a knob. To be able to test TOW mode, boat speed must be somewhat dependent on engine rpm. To achieve this, a simulator is constructed in the program. It is made as a separate entity working independently from TOW mode. Through this, engine rpm is connected to boat speed by a simple linear conversion, using the same conversion factor as the regulator (see section 4.1). So far, the simulator would not be able to test the regulator as such, since it acts on rpm alone, only the throttle response. A random number generator is added on top of the rpm to speed conversion to complete the simulator. Normally a random number generator is seeded by the system time, here however, the libraries for accessing system time, as well as those for random number functions are unavailable, so some support code to accomplish this is built into the simulator.

For debugging purposes, a developer would normally use a special device known as a debugger - essentially a special cable used to connect directly to a HCU - and some special software for reading signals, setting break points in the code and so on. Due to lack of resources, CPAC Systems is unable to provide debuggers for all developers. To circumvent this problem, a tool for debugging via system output is developed. There are a number of LED lights only any HCU, used for warnings, indicating that a certain mode is turned on, and so forth. These can be hijacked to use as indicators that certain points in the code are reached. On the side-mount HCU in particular, these LED's are arranged in an array of four lights, a simple function is created that uses these four lights to represent four bit numbers, from -15 to 15. By clever scaling this can be used to check important values.

6 Testing and verification

To verify that the systems works as intended, extensive testing i needed. As mentioned in section 5 no real boat is available during development, instead

suitable hardware is installed in a mock up rig with a simulator for the engine and a series of variable resistors representing the sensors. This rig is used for testing during development and implementation. Continuous testing during development ensures that all parts function properly. As soon as a unit has been completed, further testing verifies that everything works as specified in section 2. If any faults are found, they are corrected and the unit is tested again to make sure that no other faults have arisen with the corrections. Any faults that occur in previously tested and verified units as a result of new development are corrected on the spot but not completely tested until final system test.When all units have been implemented a final test in the rig is done to verify overall functionality and that the system adheres to specifications.

The first part of TOW mode to be implemented is the new menu system with surrounding HMI. A test protocol for menu navigation is set up and used as a base for testing. The final test protocol, after fault correction is as follows:

THING TO TEST	RESULT	COMMENT
TOW screen in menu (requirement 1d)	OK	
TOW screen in menu shows "—" if TOW is inactive	OK	
Sub-menu reachable from TOW screen	OK	
If default is 'speed' then the speed screen is reached in sub-menu	OK	
If default is 'rpm' then the rpm screen is reached in sub-menu	OK	
Sub-menu only has speed and rpm screen	OK	
On/Off screen can be reached from sub-menu	OK	
In On/Off screen, TOW mode can be activated and deactivated (requirements 2 & 3)	OK	
+/- buttons work in On/Off screen if TOW is active (requirement 8f)	OK	
If speed mode is active, rpm menu is not reachable (requirement 8a)	OK	
If rpm mode is active, speed menu is not reachable (requirement 8a)	OK	

If TOW mode is active then back button leads directly from On/Off screen to main menu	OK	
If TOW is active then OK button leads from the main menu directly to On/Off screen	OK	
If TOW is active then main menu screen shows value and unit	OK	
If speed mode is active, main menu unit is “kts”	OK	
If rpm mode is active, main menu unit is “rpm”	OK	
If TOW is inactive and side-mount button pushed then view is shifted to On/Off screen for TOW	OK	Pop-up works
If default is ’speed’ and side-mount button pushed then the screen shifts to speed screen	OK	
If default is ’rpm’ and side-mount button is pushed then the screen shifts to rpm screen	OK	
If TOW is active and side-mount button pushed, then no shift	OK	Pop-up works
If in sub-menu and side-mount button pushed then go to On/Off screen corresponding to sub-menu	OK	
TOW default should exist in settings menu	OK	
TOW default should be set to ’speed’ at first start-up	OK	
System remembers default value from start-up to start-up	Not yet implemented	How to deal with storage of settings?
TOW default can be changed by pushing OK button	OK	
A change in default actually effects all default movements	OK	
+/- buttons on side-mount only work when TOW mode is active (requirement 8f)	OK	

Once the HMI works as it should, the outer and inner state machines are implemented and tested. Since the menu system is more or less independent from the state machine, testing is done using the LED output mentioned in section 5, to show what state is currently running. A proper test protocol is deemed unnecessary as the complexity is less than that of the HMI, suffice it to say the state machines worked as expected. A more thorough test following a proper protocol is done as part of final testing.

Before the regulator can be tested, the simulator needs to be implemented and working. This is done in two steps, first implementation and test of the rpm to speed conversion then implementation and test of the disturbance. These are made separate so that simulator can be run without disturbance if necessary. Once again, there is no need for proper testing protocol, a simple observation is sufficient. After some fault correction, both rpm to speed conversion, and disturbance work as intended.

Having a working human machine interface, a state machine running smoothly in the background and a simulator responding to engine rpm with boat speed, it is time to implement and test the speed regulator. Initial tests show that the regulator is working, though be it somewhat less than optimally, however the regulator effects the state machine adversely so that the inner state machine is stuck in the engaged state, unable to disengage. Once this fault has been corrected, the regulator is optimized according to the design of section 4.2 and tested again. There are two parts of the test, stabilization, and error compensation. The way the stabilization test works is that a certain set point is chosen while the regulator is disengaged, the lever is then brought forward until the regulator engages, if the speed stabilizes at a value coinciding with the set-point, within a reasonable amount of time, the test is passed.

Once the speed has stabilized, the error compensation test can start. This test utilizes the way the speed simulator works; the output value from the simulator enters the CAN network by hijacking the signal from one of the speed sensors, the other speed sensors are unaffected. The mode switch normally used for altering between different sensor signals on the mock sensor (variable resistor pack), can now be used for switching between simulator mode, or using the variable resistor as a sensor. Having the variable resistor tuned to the set point at the start of the test, the switch is flipped, when turning the resistor down or up, the engine rpm should respond immediately. Since there is no response from the speed (unless the tester is extremely good at controlling the resistor), the engine rpm should stabilize at a value determined by the controller gain. Both of these tests are done for a large amount of set points to ensure that the system works for the entire set-point range. Tests reveal that error compensation yields incorrect values for certain set points, once this has been corrected, the regulator works, but one more test is necessary.

The final test for the regulator is the so called flow test. This is to ensure that the set point can be changed while regulator is engaged. Starting at the lowest possible set point, the lever is moved forward until the system is engaged

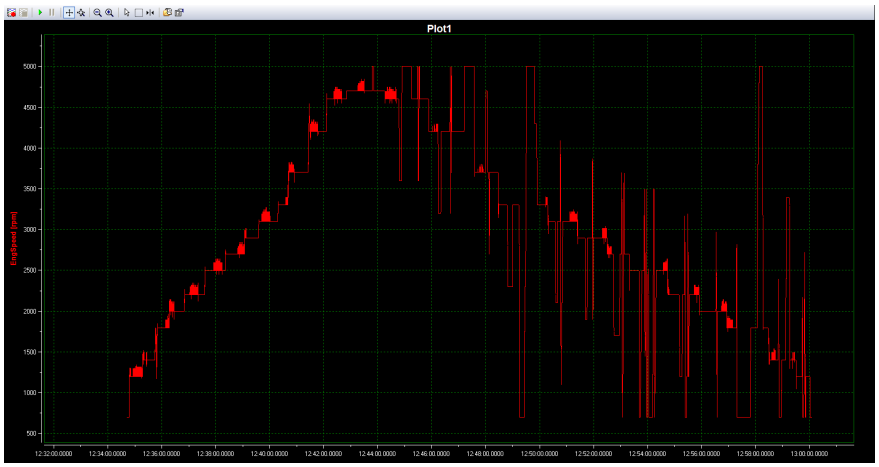

Figure 17: *Plot of the control signal from the flow test*

then on to full throttle. To be able to better monitor the changes, a computer is used to read the engine rpm via CAN bus and continually plot it on the screen. The set point is now stepped up through the entire range of set-points, observing the rpm changes and checking that the speed stabilizes correctly at certain intervals. When the maximum allowed set-point is reached, the process is reversed, stepping down the set point and observing the behavior using the plot. On the way down, error compensation tests, using the variable resistor, are done at intervals, to ensure that this still works with a flowing set-point.

The program used for data recording is unable to read the output signal but, the control signal for the flow test has been plotted infigure 17. Looking at the plot one can see a short oscillation at each level on the way up. This corresponds to the controller reacting to the small disturbances generated by the speed simulator, once the disturbances stabilize, so does the control signal. So far so good, what is more interesting is stepping down through the different set points. Here the graph fluctuates wildly. This is due to the large set-point errors introduced via the variable resistor. It is clear that the regulator reacts violently to these error signals and had the output responded to the control, the signals would have fluctuated much less. The general conclusion is that the regulator works well even if set-point changes are made while the system is running.

Final testing is comprised of a test of the HMI, a test of the state machine and a test of the regulator as well as a test of the rpm limiter to make sure that the function still works after implementation of new components. Appendix A shows the protocol from the final rig test. From this we can draw a number of conclusions, first of all that, as far as it is possible to test in the rig, all functions of TOW mode are verified. Secondly, several problems were discovered that should be corrected for optimal functionality but they are not of any major

concern, therefore TOW mode can still be regarded as functioning. Thirdly, since all tests were passed despite several problems (all though minor, they are still problems), the test procedure must be regarded as too obtuse and should be refined for future testing.

7 Further development

During the testing process, several minor problems were discovered such as problems with the read out, as well as some functions that were not implemented. In the future, correcting the problems that were found and implementing such features as were missing would be a natural path for development. As seen in section 6, the testing procedure was a bit crude as the system passed the test despite the minor problems. Defining better test cases and devising a proper test protocol that will catch these problems in the development stage is also a major goal for further development. This would have to include some sort of solution for better recording and verifying the output from the system when the regulator is engaged.

The controller parameters, as well as those for the speed simulator are all based on some assumptions that fit well with the values from the rig in the office. These parameters are at the moment hard coded into the system. Being able to adjust these parameters on the fly or even estimating them automatically would be of great benefit. The reason this has not been included is that it would go beyond the scope of the project. After minor flaws have been corrected, such functionality should be the main goal of any further development.

8 Results

We now need to determine whether or not we fulfilled the goals of the project (section 1.2) by looking at the problem definition (section 1.1) as well as the specification (section 2) and comparing with our results. The controller, consisting of a regulator for the boat speed along with a support structure in the form of a human machine interface and a state machine, has been developed and implemented within the existing hardware and software framework, all according to the goals. Tests (section 6) verify that the system fulfills the requirements of section 2.1 both concerning the state machine and the HMI. Furthermore, tests show that the regulator functions properly during simulation in the rig and, assuming that the hardware in the rig corresponds to that of a real boat, the system should work in the boat as well. However, as mentioned in section §7, some further development is needed before TOW mode can be released for use.

References

[Browning] Browning, A W: A Mathematical Model to Simulate Small Boat Behavior, *Transactions of the society for modeling and simulation international*, **56**:329, May 1991.

[Gerr] Gerr, D: *Propeller Handbook: The Complete Reference for Choosing, Installing and Understanding Boat Propellers.* London: Nautical Books, 1989.

[Lennartson] Lennartson, B: *Reglertekniens grunder.* Lund: Studentlitteratur, 2002.

[Phillips-Birt] Phillips-Birt, D: *Motor Yacht and Boat Design.* London: Adlard Coles Limited, 1966.

[Teale] Teale, J: *High Speed Motor Boats.* Lymington, Hampshire: Nautical Publishing Company, 1969.

[Tornblad] Tornblad, J: *Fart och propeller för motorbåtar.* Kristinehamn 1990.

[Xi & Sun] Xi, H & Sun, J: Feedback Stabilization of High-Speed Planing Vessels by a Controllable Transom Flap, *IEEE Journal of Oceanic Engineering*, **31**:2, April 2006

[Åström & Wittenmark] Åström, J K & Wittenmark, B: *Computer Controlled Systems. Theory and Design. 3rd ed.* Upper Saddle River: Prentice hall, 1997.

A The final rig test protocol

A.1 Human machine interface test

THING TO TEST	RESULT	COMMENT
TOW screen in menu (requirement 1d)	OK	
TOW screen in menu shows ”—” if TOW is inactive	OK	
Sub-menu reachable from TOW screen	OK	
If default is 'speed' then the speed screen is reached in sub-menu	OK	
If default is 'rpm' then the rpm screen is reached in sub-menu	OK	
Sub-menu only has speed and rpm screen	OK	
On/Off screen can be reached from sub-menu	OK	Checked for both rpm and speed
In On/Off screen, TOW mode can be activated and deactivated (requirements 2 & 3)	OK for speed OK for rpm	
+/- buttons work in On/Off screen if TOW is active (requirement 8f)	OK for speed OK for rpm	
If speed mode is active, rpm menu is not reachable (requirement 8a)	OK	Switching the default does not change this
If rpm mode is active, speed menu is not reachable (requirement 8a)	OK	Switching the default does not change this
If TOW mode is active then back button leads directly from On/Off screen to main menu	OK for speed OK for rpm	
If TOW is active then OK button leads from the main menu directly to On/Off screen	OK	Checked together with previous
If TOW is active then main menu screen shows value and unit	OK	
If speed mode is active, main menu unit is “kts”	OK	

If rpm mode is active, main menu unit is “rpm”	OK	
If TOW is inactive and side-mount button pushed then view is shifted to On/Off screen for TOW	OK	
If default is ’speed’ and side-mount button pushed then the screen shifts to speed screen	OK	Somewhat faulty Value, fix later
If default is ’rpm’ and side-mount button is pushed then the screen shifts to rpm screen	OK	Somewhat faulty Value, fix later
If TOW is active and side-mount button pushed, then no shift	OK	
If in sub-menu and side-mount button pushed then go to On/Off screen corresponding to sub-menu	OK for speed OK for rpm	Value switching seems disturbed by side-mount button
TOW default should exist in settings menu	OK	
TOW default should be set to ’speed’ at first start-up	OK	
System remembers default value from start-up to start-up	Not yet implemented	How to deal with storage of settings?
TOW default can be changed by pushing OK button	OK	
A change in default actually effects all default movements	OK	Tested in earlier steps
+/- buttons on side-mount only work when TOW mode is active (requirement 8f)	OK	Slightly too sensitive, needs tuning

A.2 State machine test

(

Thing to test	Result	comment
System is unavailable if parameter not set (requirement 1d)	Unable to test	Parameter always set
TOW mode can only be activated if lever is neutral (requirements 2c & 3c)	OK	Causes problem in menu system, to be fixed later.

If activate is requested and lever is not neutral, deny activation	OK	Same test as previous, might need to be indicated in HMI
If speed mode requested, activate speed mode	OK	
If rpm mode is requested, activate rpm mode	OK	
Activation goes via activation in progress state for both rpm and speed mode	OK for speed OK for rpm	
Only one mode active at a time, either speed or rpm (requirement 8a)	OK	
Set point or rpm limit can only be changed if system is active (requirement 8f)	OK	Tested in menu system test
System can only be deactivated if lever is neutral (requirement 4b)	OK	Tested for both speed and rpm. No menu problem here
If deactivation requested and lever is not neutral, deny deactivation	OK	Same test as previous test, indicate to user?
Deactivation goes via deactivation in progress	OK for speed OK for rpm	
If other function disables, deactivate	OK for speed OK for rpm	
Deactivation after other function disables goes via abandoned	OK	Check for both speed and rpm
Engage is only possible if system is active (requirement5a)	OK	
Only the active mode (speed or rpm) can be engaged (requirement8b)	OK	
Only engage in forward gear (requirement 5b)	OK	
Engage if rpm is equal to or above set-point/limit (requirement 5c)	OK for speed OK for rpm	
Disengage when rpm is lower than set-point/limit (requirement 6b)	OK	

In speed mode, regulate boat speed when engaged (requirement 7a)	OK	
In rpm mode limit rpm when engaged (requirement 7b)	OK	
If engaged and set-point/limit is changed, adjust rpm accordingly immediately (requirement 7c)	OK for speed OK for rpm	
When engaged: if set-point/limit is raised above current rpm, disengage (requirement 7d)	OK for speed OK for rpm	
When disengaged, throttle responds to user command like normal (requirement 8c)	OK	
As long as system is engaged, lever position is not taken into consideration other than for disengaging. (requirement 8d)	OK	
System can only engage when lever is forward, in reverse gear, respond like normal to throttle position. (requirement 8e)	OK	

A.3 Regulator test

SET-POINT (KTS)	STABILIZATION	ERROR COMPENSATION	COMMENTS
5	OK	OK	
7	OK	OK	
9	OK	OK	
11	OK	OK	
13	OK	OK	Unstable if engaged With ignition off
16	OK	OK	Stabilizes to too high value at times
18	OK	OK	
20	OK	OK	
22	OK	OK	Faster stabilization for values over 20 kts
24	OK	OK	
27	OK	OK	

29	OK	OK	
30	OK	OK	
33	OK	OK	
35	OK	OK	
38	OK	OK	
40	OK	OK	40 is slightly too high when using the rig, variable resistor gives error on higher values