Sailboat Weaving: Racing Strategies in a Wind Vector Field

Christopher K. Lee
Applied Mathematics 115
December 12, 2010

Abstract

The goal of this model is to study the best point-of-sail; particularly, whether the best strategy for a sailboat is to hold a constant course or to weave through a course during gusts and lulls. Methods in vector calculus will utilized for basic assumptions in wind vector fields and paths of boats in the field. Modeling boat vector paths can be tested with actual wind conditions and may lead to new knowledge in sailing technique.

Basic Sailing Theory

When considering the physical principals of sailing, it is important to understand and contextualize the importance of the point-of-sail in reference to wind direction and strength. Moving air flowing past a sail provides lift, which produces an overall force in the direction of motion (Fig. 1). Counter-intuitively, the fastest point-of-sail is not sailing downwind because in this instance the boat is pushed from behind and will never reach wind velocity due to drag. Sailing close into the wind at a 41-45° angle produces the fastest boat velocity. Some boats can reach speeds up to 3x the wind speed in this point-of-sail (close hauled).

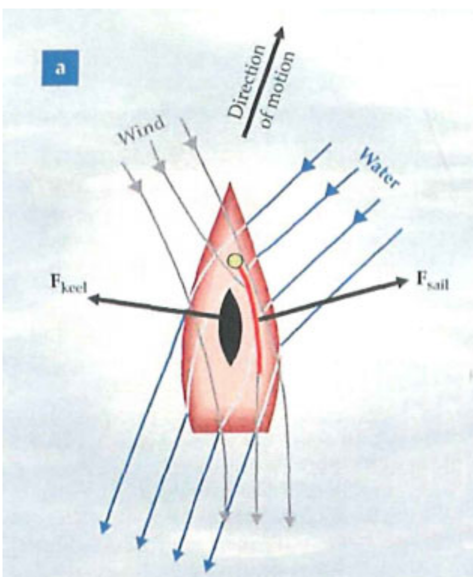

Figure 1a

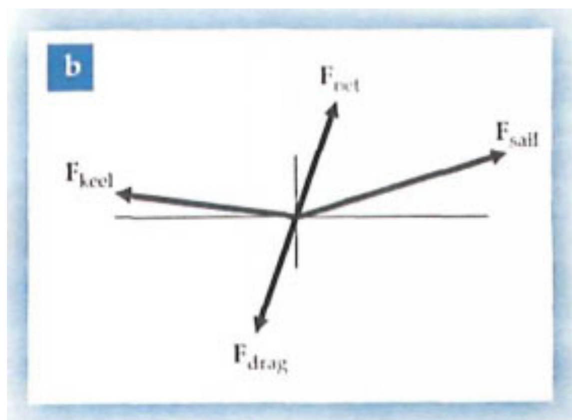

Figure 1b

Figure 1a *illustrates the forces that act on a moving sailboat.* ***Figure 1b*** *shows that the vector sum of the lift force and the keel force and negative drag force determines the net force. The position, direction and velocity of a boat can be determined and modified with vector addition (Physics Today, 2008).*

The goal of collegiate sailing is to set a heading that will allow you to finish the course first. The challenge begins even before the race starts because there is always a more favorable position at the starting line. Once the horn is blown, the next challenge is to navigate the boat by zigzagging left and right (tacking) toward the windward mark. In theory, wind is constant from the “windward” mark, and theory says sailing closest to the wind is the fastest course to the mark, but not necessarily the fastest velocity. However, in reality, the wind is not constant and does not come from a single direction, but instead wind comes in gusts and lulls that can also shift in direction. Good sailors are said to be able to "read" the water by identifying patches of dark water where these gusts disturb the water thereby allowing them to predict when the gust will hit. This wind behavior can be represented with a wind vector field or a wind contour map generally usually used at a macro-scale for atmospheric study. Wind vector data at a micro-level do no exist because satellite imagery is not sensitive enough, though the same macro principles can be applied in creating an artificial field.

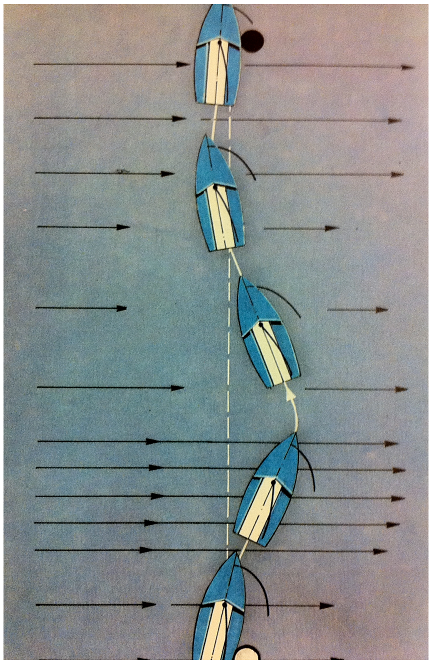

Figure 2

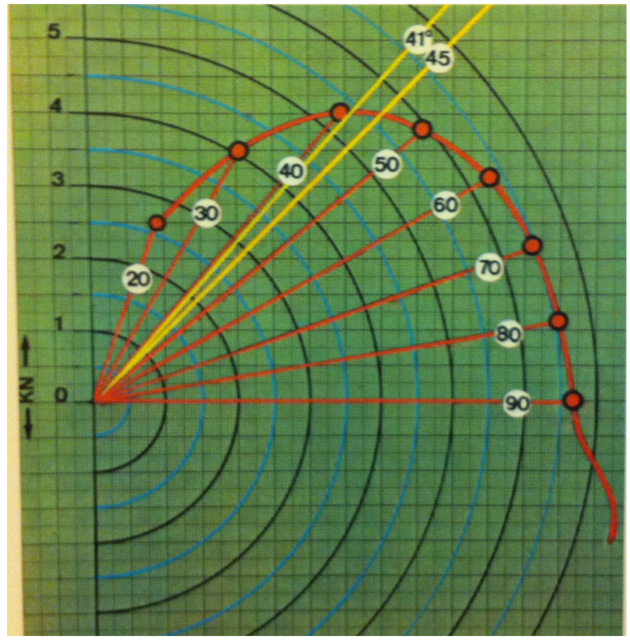

Figure 3

Figure 2 *demonstrates how weaving through the gusts and lulls produces the fastest reaching course. The general rule is to bear away from the gusts (turn away from the wind) and to luff in the lulls (overturn into the wind) rather than to stay a straight course on the dotted line. This wind behavior forms a wind contour that can be represented as a wind vector field (Bavier, 1945).* ***Figure 3*** *plots boat angle and boat speed on a graph. The wind direction is parallel to the y-axis, and the straight red lines represent the boat's heading. The alternating blue and black semi-circles demonstrate speed. The optimum speed threshold is a heading between 41° and 45° to the true wind (Imhoff & Prange, 1975).*

All of the aforementioned will be taken into consideration when constructing this model. Deciding what angle to sail, how many tacks to take, whether to sail on the port or starboard tack (left or right side), how close to the wind you want to sail, how to distribute weight, how to modify the shape of the sail, how to read the apparent and true winds, and more are all complications that will be left out of this model to distill the question at hand.

The Question Framed

Given a wind front that is uni-directional, which of the following two sailing strategies perform best in gusts and lulls?

1) Sail close to the wind no matter what to achieve the shortest and straightest course length
2) Weave through gusts and lulls creating a longer course, but maintaining a higher velocity

Structuring the Model

As illustrated in Figure 4, this model will only study the last leg of the course as it approaches the mark. In other words, the boat will be on its last tack and will have carried velocity and acceleration from its previous tack. The decision to structure the wind so that it flows eastwards as the boat sails northwards is to create a perpendicular axis so that resultant forces are more clearly illustrated. This point-of-sail is called *broad reach*, and it has been chosen because it is safe to assume that most sailors point their sails at a 45° angle away from the true wind (β angle). The Figure 1 illustrates a more commonly used point-of-sail called *closed-hauled*, but this point of reference will make calculating force vectors and angels altogether too confusing.

It is important to note, however, that the underlying principles of weaving through gusts and lulls in a broach reach (Figure 2) can be applied to feathering strategies and Ease-Hike-Trim strategies at other points of sail. All incorporate an oscillating technique.

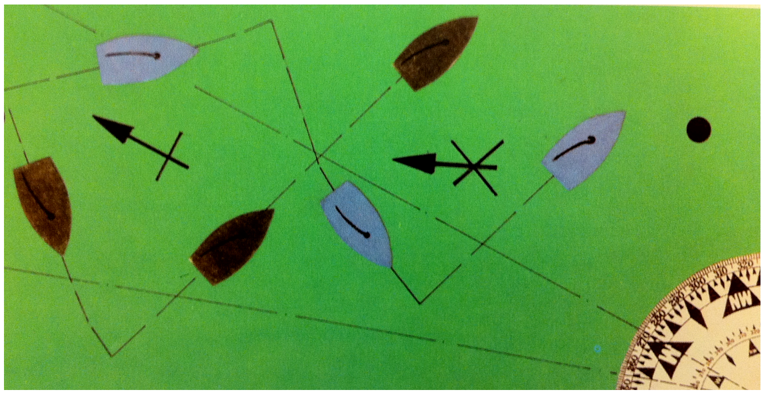

Figure 4

Figure 4 *demonstrates how a simple wind-shift puts the blue boat closer to the mark, even though the brown had the advantage at the start. Discerning the favorable tack, judging potential wind shifts, and deciding when to change course are complexities that make sailboat racing exciting (Fisher, 1976).*

Artificial Wind Vector Field

The artificial wind vector fields are can be set up with the following two the arbitrary equations:

```
Z=(-0.5)*abs(5-Y)/5+(X).^(1/1.6)+cos(X./2);
Z=(-0.5)*abs(5-Y)/5+(X+5).^2;
```

We then calculate gradient vector in its U and V components, so that we visualize the magnitude of the wind vector at a certain point.

```
[FX, FY] = gradient(F,h)
```

$$\nabla F = \frac{\partial F}{\partial x}\hat{i} + \frac{\partial F}{\partial y}\hat{j}$$

The prepared vectors are plotted on a mesh-grid that measures 11x13 units for easy illustration of 143 vectors. The varying magnitude of each vector corresponds to the strength of the wind at a given grid point, which then creates gusts and lulls at certain parts of the course. Sailing through a gust wind vector means that the boat will accelerate, and catching a lull wind vector means deceleration.

Wind Vectors to Boat Forces

When close-hauled, the **boat** heads between 40° and 50° to the true wind. At broad-reach, the **sail** is pointed between 40° and 50° away from the true wind. In either case, there is an angle threshold no matter the wind direction, boat direction or sail direction.

As mentioned before, rather than describing motion from the reference point of the boat, and thinking of the true wind angle, the apparent wind angle, and the sail angle in relation to the boat direction, I have decided to peg the incoming-wind-to-sail angle to be 45°, a general assumption for optimum speed to windward. Using this reference point will assume that at any point-of-sail and given wind direction, the sailor is intelligent enough to adjust his boat to achieve this optimum speed threshold. This concept may take some time to digest, but reviewing the figures above will reaffirm this relationship.

With this notion in mind, when looking at an arbitrary vector field, it does not matter what directions the wind vectors are pointing—what does matter are the magnitudes of the vectors. The pegged-45°-relationship will break down the wind vector into its cos(45) and sin(45) components and will be applied to the boat's forces (`Fsail+Fkeel+Fdrag=Fnet`) through vector addition. Changes in net force can then be calculated into boat velocity and acceleration with Newtonian mechanics. Given force = mass*acceleration, these forces can be converted acceleration (dx2/dt & dy2/dt) and integrated into velocity (dx/dt & dy/dt) and then decomposed to its X and Y components to describe the position of the boat.

```
F=ma
a=F/m
d=vt+(1/2)*at^2
```

Sailing through Wind Field

Directing the walk through the field proved to be the most challenging aspect of this model. A boat would move from one point to the next in the field through vector addition. I realized that I had constructed something similar to a robot moving through a grid, taking instructions at a grid point, and moving on to the next point. However, in this instance, the instructions are the wind vector force components, and the next point is the vector sum of Fsail, Fkeel and Fdrag.

The force of the wind acting are converted into Fsail, Fkeel and Fdrag:

```
tempFsail=[abs(cos(45)*Fwind(2)*(dist))*dt, abs(sin(45)*Fwind(2)*dist)*dt];
tempFkeel=[-abs(cos(45)*Fwind(2)*(1-dist))*dt, abs(sin(45)*Fwind(2)*(1-dist))*dt];
tempFdrag=[abs(cos(45)*Fwind(2)*0.15)*dt, -abs(sin(45)*Fwind(2)*0.15)*dt];
```

Absolute values are used so that when a boat sails into a wind (close-hauled), the wind does not push it backwards, but actually propels the boat forwards due to the explanation of lift before. If the wind is approaching perpendicularly (broad-reach) or from behind (running), the absolute value converts a vector into positive propulsion. Though it is clear that this is an unpolished interpretation of fluid dynamics, the underlying principles governing sailboat movement are implemented. A factor of 0.15 is multiplied to drag to take into consideration the rudder drag as it tries to maintain a course. A distribution factor dist=0.8 is used to show that Fsail contributes to the net forward movement more so than does the Fkeel. Negative values are used to appropriate the correct directions of the vectors.

Key Functions

- *race.m* – master function, initializes subplots, axis, sailboat.structure, windfield, and animation. Takes in parameters boat starting position x,y, starting Fsail [u v], starting Fkeel [u v], time step dt, width and height of the course in meters, choice of vector field n, and choice of course b.

    a. *init.m*– initializes sailboat structure. The desired mass of two sailors is 127kg, and the average hull mass of a FJ Dinghy is 95kg. Sample structure data after one experiment run:
        - `boat =`

```
                x: 0.2346
                y: 9.9717
             mass: 222
            Fdrag: [0.3728 -1.6039]
            Fsail: [-0.7529 4.7206]
            Fkeel: [0.7529 4.7206]
             Fnet: [0.3728 7.8373]
         velocity: [0.2759 8.2663]
     acceleration: 0
            speed: [1x328 double]
            pathx: [1x327 double]
            pathy: [1x327 double]
```

b. *windfield.m*– plots the course, calculates gradients, and renders individual wind vectors. It is important to note that the actual meshgrid dimensions used to calculate sailboat movement are 110x130, giving 14300 grid-points. This higher resolution allows sailboat movement to be more accurate during the weaving strategy. Otherwise, in an 11x13, 143 grid-point resolution, the density is too low and, the boat just sails down a single row (row 6) of the grid-points. In the higher density, the boat fluctuates between rows 55±15 out of a total of 110 rows. The lower 11x13 resolution and is rendered because it has a lower density and the individual wind vectors are more easily visualized. The creation of the meshgrid:

```
XX = linspace(-x,x,110);       % e.g. 110 gridpoints in x
YY = linspace(-1,y,130);         % e.g. 130 gridpoints in y
[X,Y] = meshgrid(XX,YY);
```

- *windfunc.m*– allows the user to choose between equations drawing the wind vetor field. Equation 1 works best, equations 3&4 have been commented out. I could not figure out what equation could render gusts and lulls seen in Figure 2.

c. *animate.m*– modeled after move.m in from lecture. Transitions boat.structure per timestep dt.

- *sailing.m*– main engine calculating path &velocity of boat sailing through field. Important parts of the code include 1) calculating the index for [U,V] given a the (x,y) position of a boat, 2) converting force into the boat velocity, 3) calculating the boat position given a straight-course strategy vs. weaving strategy, and 4) calculating speed by finding the differential of path position for the validation of model

  1) *Fwind Index*, returns values between 1:14300 for wind vector componenets [U(i),V(i)]

```
i=(round(55*boat.x+55))*130+round(boat.y*13)+1;
u = U(i);
v = V(i);
Fwind = [u v];
```

  2) *Velocity Calculation*, modeled after move.m

```
boat.velocity=[boat.velocity(1)+dt*boat.acceleration+accelx
, boat.velocity(2)+dt*boat.acceleration+accely];
```

  3) *Position Calculation*, verbatim from move.m for straight-course

```
if b==1
    %straight-course strategy
    %the following two lines are taken from move.m
    boat.x=boat.x+0.5*dt*boat.velocity(1);
%d=(1/2)*at^2=(1/2)*v*t
    boat.y=boat.y+0.5*dt*boat.velocity(2);
    %weaving-course strategy
elseif b==2
    boat.y=boat.y+dt+dt*0.5*boat.velocity(2);
    boat.x=sin(boat.y*pi/1.5)/5+dt*0.5*boat.velocity(1);
end
```

  4) *Speed Calculation,* derivative of boat.path

```
dx=100*diff(boat.pathx);
dy=100*diff(boat.pathy);
D=(dx.^2+dy.^2).^(1/2);
```

In all, these functions and equations make up the boat movement model. The x and y components for all vectors are broken down and iterated when the boat moves through the field. Factors such as water resistance and terminal velocity have been left out because they depend more on hull design than they do on sailing strategy. The resultant model is an animation that can be viewed online.

`http://www.youtube.com/watch?v=Ap5XPPgefcE`

Summary of Results

The plot is held constant so that multiple boats with varying parameters can be tested in the same wind vector field. The first subplot gives the position of the boat within the vector field. The red dotted line is the desired boat path. The second subplot gives the live boat speed as the sailboat goes through its course.

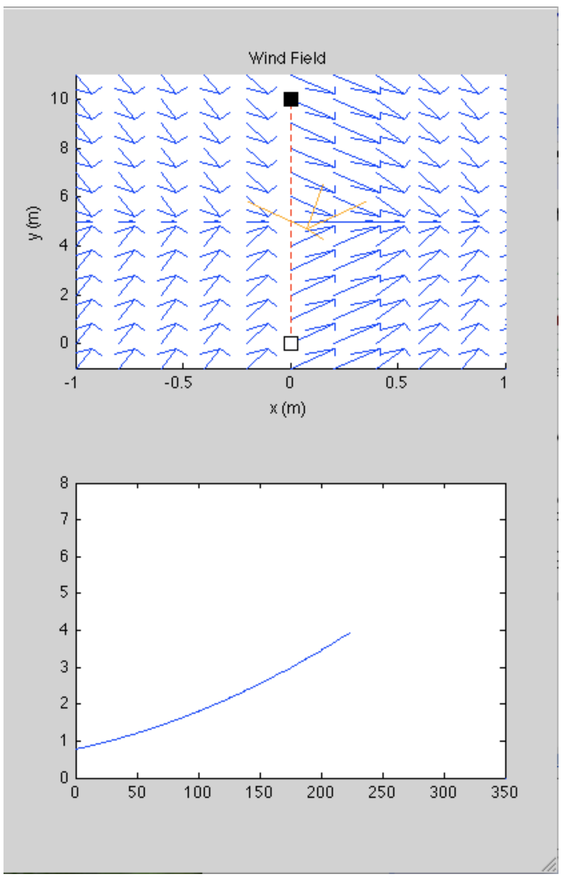

Figure 5

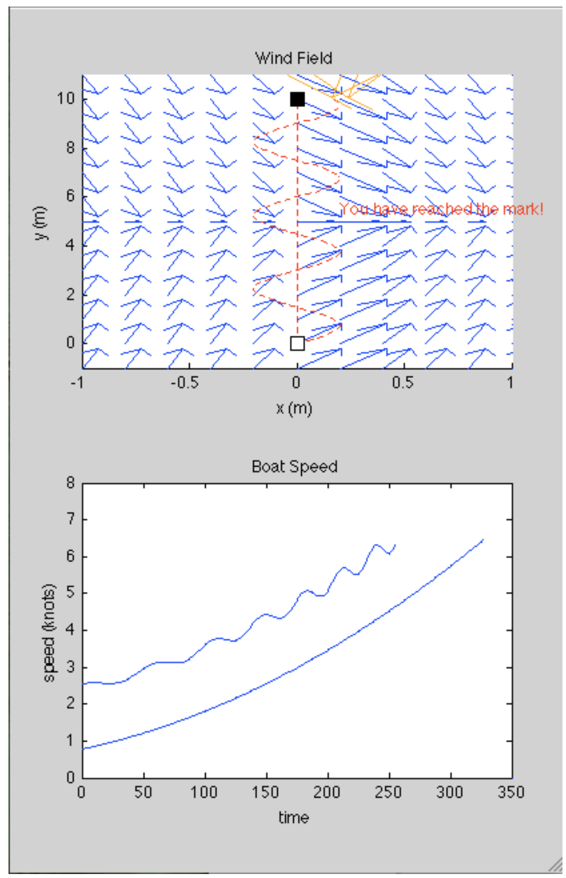

Figure 6

Figure 5 *illustrates a sailboat halfway through its course. As it moves through the wind field, its boat speed is plotted as a function of time. The position of the boat is where the yellow vectors converge. The magnitude and direction of the four vectors* (`Fsail+Fkeel+Fdrag=Fnet`) *change as the boat encounters different wind vectors. The resultant drag vector is pushing the boat slightly towards the east.* ***Figure 6*** *demonstrates the completion of two sail strategy testing. The weaving path follows the sinusoidal path, and starts with a greater initial velocity because the boat catches the first gust from the start. Note that the weaving strategy completes this leg of the race in less time.*

Actual wind data was not used because it is not practical to take a single wind velocity and direction from one measuring instrumentation position (usually at a boathouse pavilion), and used to create a field of wind vectors from that. The fact that the weaving-strategy is faster than the straight-course strategy validates the general advice given by experienced sailors, "Look for gusts and lulls: bear away in gusts to stay in them longer and luff in the lulls to increase speed and head towards the next gust." Given the above parameters, results and modified iterations, the average advantage of weaving over straight is given by: finishing the course in $6/7^{th}$ of the time, a 1.42 meters lead, or a 1/3 boat-length advantage on a 10-meter leg. Though I could not find a number in the sailing literature to validate these figures, I emailed my coach about this, and he said it sounded about right. This is crucial in fleet racing, where a leading boat will leave behind a wake of wind pollution, and pursuing boats have less wind to catch up.

Closing Remarks

There were many stages of this term project that seemed impossible. The challenge was to take a very large and general problem and reducing it down to smaller, solvable problems: how to sail faster; what two strategies to test; what point-of-sail do I choose; how do I factor in the angles of everything; how to best normalize these factors; how to create a wind vector field; recognizing the importance of a meshgrid; how to calculate a wind force into a new boat position; how to factor in vectors for boat movement; how create a desired boat course; and, how to compare the results of the two strategies. Though I have logged 36 hours on this term project, I have found this endeavor very rewarding, and did not realize the full potential of modeling, until I pursued a topic I am interested in and familiar with.

I would like to thank my roommate Ben Li, a physics concentrator, for helping me with the vector addition. I recognize that the pegged-45°-relationship is a crude assumption, but it is difficult for me to gauge how accurate it is. One possible improvement is factor in true wind, the apparent wind and the point of sail to give a better approximation of boat movement.

Finally I would like to thank Professor Bossert and Gang Yu for being patient and giving initial direction in this project. Since coming to Harvard, I have taken two computer science courses and two proof-based mathematics courses, and am concurrently enrolled in Applied Mathematics 21a. I look forward to learning more mathematical concepts more in depth so that they can be applied.

Appendix

Resources

Anderson, Byron D. "The Physics of Sailing." *Physics Today* February (2008). Web. 9 Dec. 2010.

<http://isites.harvard.edu/fs/docs/icb.topic819458.files/Lift/PhysToday.pdf>.

Bavier, Robert Newton. *Sailing to Win*. New York: Dodd, Mead, 1978. Print.

Fisher, Bob. *Crewing Racing Dinghies and Keelboats*. New York: Dodd, Mead, 1976. Print.

Imhoff, Fred, and Lex Pranger. *This Is Boat Tuning for Speed*. New York: Hearst Marine, 1985. Print.

Matlab Functions

- race.m
- init.m
- windfield.m
- windfunc.m
- animate.m
- sailing.m

race.m

```
function [boat]=race(x,y,Fsail,Fkeel,dt,width,height,n,b)

%initialize sailboat
[boat]=init(x,y,Fsail,Fkeel);

subplot(2,1,2)
title('Boat Speed'), xlabel('time'), ylabel('speed (knots)')
t0=350;
plot(t0,0);
axis([0 t0 0 8]);
hold on

subplot(2,1,1)
title('Wind Field'), xlabel('x (m)'), ylabel('y (m)')
%render field
[U,V,u,v]=windfield(width,height,n,b);
hold on

%animate sailing
[boat] = animate(boat,dt,width,height,U,V,u,v,b);



end
```

init.m

```
function [boat]=init(x,y,Fsail,Fkeel)
%Initialize the boat data structure
%   function boat = initsail() takes in parameters:
%
%   x is the x position in field
%   y is the y position in field
%   mass is the combined mass of boat
%   Fdrag is drag force vector in [u v]
%   Fsail is sail force vector in [u v]
%   Fkeel is keel force vector in [u v]
%
%   Fnet is the net force vector in [u v]
%   velocity is the magnitude of net force vector
%   acceleration
%
%   speed stores the boat's velocity
%   path.x stores the x path of boat
%   path.y stores the y path of boat


boat.x=x;
boat.y=y;
boat.mass=127+95;           %avg mass of crew=127kg, hull=95kg

Fdrag = [0 -1];
%Fdrag(2) = -(Fsail(2)+Fkeel(2))/10;
boat.Fdrag=Fdrag;
boat.Fsail=Fsail;
boat.Fkeel=Fkeel;
boat.Fnet=Fsail+Fkeel+Fdrag;

boat.velocity=[0 1];            %boat has velocity from last tack
boat.acceleration=0;      %boat speeds up slightly after a good roll-tack

boat.speed=[];
boat.speed=[boat.speed boat.velocity];

boat.pathx=[];
boat.pathy=[];
boat.pathx=[boat.pathx x];
boat.pathy=[boat.pathy y];
end
```

windfield.m

```
function [U,V,u,v] = windfield(x,y,n,b)
%Wind vector field.
%   windfield(x,y) sets up the grid structure for the wind vector field
%

%For visualization 11x13 grid

%First define grid in which boat will be sailing through
XX = linspace(-x,x,11);      % e.g. 11 gridpoints in x
YY = linspace(-1,y,13);        % e.g. 13 gridpoints in y
[X,Y] = meshgrid(XX,YY); % Sets matrix for grid system in x and y
axis([-1*x 1*x -1 y]);
hold on
[Z]=windfunc(X,Y,n);
%[X,Y,Z] = peaks(XX);
%[C,h] = contour(X,Y,Z,10);
[U,V] = gradient(Z,.1,.01);
U=real(U);
V=real(V);
v=quiver(X,Y,U,V);
colormap winter
set(v,'Color','b');
%set(h,'ShowText','on','TextStep',get(h,'LevelStep')*5)
hold all

%

%First define grid in which boat will be sailing through
aXX = linspace(-x,x,110);      % e.g. 11 gridpoints in x
aYY = linspace(-1,y,130);        % e.g. 13 gridpoints in y
[aX,aY] = meshgrid(aXX,aYY); % Sets matrix for grid system in x and y
[aZ]=windfunc(aX,aY,n);
[u,v]=gradient(aZ,.1,.01);
u=real(u);
v=real(v);
[u,v];


%Set up x & y to draw line for a straight course
y = 0:0.1:10;

if b==1
    x=zeros(1,101);
elseif b==2
    x=sin(y*pi/1.5)/5;
end

plot(x,y,'LineStyle', '--','Color','r');
plot(0,0,'--rs','MarkerSize',10,'MarkerFaceColor','w',...
    'MarkerEdgeColor','k');
plot(0,10,'--rs','MarkerSize',10,'MarkerFaceColor','k',...
    'MarkerEdgeColor','k');

end
```

windfunc.m

```
function [Z]=windfunc(X,Y,a)
% wind function to create artificialwind vector field
%

if a==1
    Z=(-0.5)*abs(5-Y)/5+(X).^(1/1.6)+cos(X./2);
elseif a==2
    Z=(-0.5)*abs(5-Y)/5+(X+5).^2;
%{
elseif a==3
    Z=abs(Y-5)/5+sin(Y.*pi)/5+(X).^(1/1.6)+sin(X.*pi)/5;
elseif a==4
    Z=(-0.5)*Y.^(0.5)+(X+5).^(1/1.6);
%}
end

end
```

animate.m

```
function [boat] = animate(boat,dt,width,height,U,V,u,v,b)

x=boat.x;
y=boat.y;
ratio=0.3;
subplot(2,1,2)
    dx=100*diff(boat.pathx);
    dy=100*diff(boat.pathy);
    D=(dx.^2+dy.^2).^(1/2);
    length=size(D);
    t=1:length(2);
    Dt=plot(t,D);

    subplot(2,1,1);

position=plot(x,y);%,'d','EraseMode','xor','MarkerSize',15,...
    %'MarkerFaceColor','g','MarkerEdgeColor','k');

    %animate force vectors acting on boat
x1 = x+boat.Fnet(1);
y1 = y+boat.Fnet(2);
x2 = x+boat.Fsail(1);
y2 = y+boat.Fsail(2);
x3 = x+boat.Fkeel(1);
y3 = y+boat.Fkeel(2);
x4 = x+boat.Fdrag(1);
y4 = y+boat.Fdrag(2);
netforce = plot([x x1],[y y1]);
sailforce = plot([x x2],[y y2]);
keelforce = plot([x x3],[y y3]);
dragforce = plot([x x4],[y y4]);


mark = height-1;
while (y<mark)
  [boat,t,D]=sailing(boat,dt,u,v,b);
    x=boat.x;
    y=boat.y;
    t=t;
    D=D;
     subplot(2,1,2)
    set(Dt,'xData',t,'ydata',D);
    Dt=plot(t,D)
    subplot(2,1,1)
    set(position,'xData',x,'ydata',y);

    x1 = x+boat.Fnet(1)*ratio;
    y1 = y+boat.Fnet(2)*ratio;
    x2 = x+boat.Fsail(1)*ratio;
    y2 = y+boat.Fsail(2)*ratio;
    x3 = x+boat.Fkeel(1)*ratio;
    y3 = y+boat.Fkeel(2)*ratio;
    x4 = x+boat.Fdrag(1)*ratio;
    y4 = y+boat.Fdrag(2)*ratio;
    set(netforce,'xData',[x x1],'yData',[y y1],'Color',rgb('Orange'));
    set(sailforce,'xData',[x x2],'yData',[y y2],'Color',rgb('Orange'));
    set(keelforce,'xData',[x x3],'yData',[y y3],'Color',rgb('Orange'));
    set(dragforce,'xData',[x x4],'yData',[y y4],'Color',rgb('Orange'));
  drawnow

  if (x<-width)|(x>width)|(y<-1)|(y>height)
        y=height;
        text(0.2,4.5,'Out of Bounds!','Color','r');
  end
end
if y>=mark
    text(0.2,5.5,'You have reached the mark!','Color','r');
end
end
```

sailing.m

```
function [boat,t,D] = sailing(boat,dt,U,V,b)
%sailing.m is the boat movement engine

% crude estimation of wind distibution on boat forces, eta
dist = 0.8;

%calculate index for [U,V] meshgrid given the boat's [x,y] position
i=(round(55*boat.x+55))*130+round(boat.y*13)+1;
%i=(round(boat.x)+1)*13+round(boat.y)+1
u = U(i);
v = V(i);
Fwind = [u v];

%sail force
tempFsail=[abs(cos(45)*Fwind(2)*(1-dist))*dt, abs(sin(45)*Fwind(2)*dist)*dt];
%keel force
tempFkeel=[-abs(cos(45)*Fwind(2)*(1-dist))*dt, abs(sin(45)*Fwind(2)*(dist))*dt];
%drag force, mostly due to rudder to prevent slippage
tempFdrag=[abs(cos(45)*Fwind(2)*0.15)*dt, -abs(sin(45)*Fwind(2)*0.15)*dt];
%net force

tempFnet=boat.Fnet+tempFsail+tempFkeel+tempFdrag;
accelx=tempFnet(1)/boat.mass;            %a=F/m
accely=tempFnet(2)/boat.mass;
%
% After wind, if new net force is greater than old net force, accelerate
if (sum(boat.Fnet)<=sum(tempFnet))
    boat.Fnet=tempFnet;
    boat.Fsail=boat.Fsail+tempFsail;  %adding force
    boat.Fkeel=boat.Fkeel+tempFkeel;
    boat.Fdrag=boat.Fdrag+tempFdrag;
    boat.velocity=[boat.velocity(1)+dt*boat.acceleration+accelx...
        boat.velocity(2)+dt*boat.acceleration+accely]%pos accel
end

if b==1
    %straight-course strategy
    %the following two lines are taken from move.m
    boat.x=boat.x+0.5*dt*boat.velocity(1);     %d=(1/2)*at^2=(1/2)*v*t
    boat.y=boat.y+0.5*dt*boat.velocity(2);
    %weaving-course strategy
elseif b==2
    boat.y=boat.y+dt+dt*0.5*boat.velocity(2);
    boat.x=sin(boat.y*pi/1.5)/5+dt*0.5*boat.velocity(1);
end

totalspeed=boat.velocity(1)+boat.velocity(2);

%store history
boat.speed=[boat.speed totalspeed];
boat.pathx=[boat.pathx boat.x];
boat.pathy=[boat.pathy boat.y];


    dx=100*diff(boat.pathx);
    dy=100*diff(boat.pathy);
    D=(dx.^2+dy.^2).^(1/2);
    length=size(D);
    t=1:length(2);
    Dt=plot(t,D);

end
```