Modelling Assignment 1

Daniel Shor: 4745094

## Introduction to Forces

There are two forces acting on the duck. The first, is the force of gravity -- Fg(N) = M(g)\*9.8(m\*s^2). The second is the buoyant force on the duck -- Fb(N) = ρ(g/m^3)\*9.8(m\*s^2)V(m^3). ρ is the density of the object, and V is the volume of the displaced fluid (submerged volume). If the duck is to float, the force vector must be equivalent (have the same length) and opposite (have the same x-axis value). In order to determine if the duck will float, we must do the following:

1)Find the total area of the duck, 2) Find the center of mass, 3) Find the force of gravity on the duck (Fg), 4) Find the submerged area, 5) Find the center of mass for the submerged area, 6) Find the force of buoyancy (Fb)

## Total Area of the Duck

First, we must triangulate the duck. Using a custom implementation of triduck.m, I convert the shape into 8455 triangles. Delauney Triangulation was used to create a shape approximation. This was refined via the mesh2d library refine2.m file. Triangulation is a more accurate way of approximating shapes than a cubic or circular shape simplification. Using the code found in the triduck.m file, I then find the area of each individual triangle, input that into a matrix, and sum the entire matrix to find the total area of the duck – 586.6353mm^2.

## Center of Mass

Once we have the area, we must find the center of mass. That is accomplished in the triduck.m file using a loop. Each triangle’s center of area is calculated, by averaging the three x coordinates and three y coordinates that comprise the triangles vertices. The x and y coordinate of the center are then solved using the below code.

cx = sum(AreaPerTriangle.\*CenterPerTriangle(:,1))/areaduck;  
cy = sum(AreaPerTriangle.\*CenterPerTriangle(:,2))/areaduck;  
ShapeCenter = [cx cy];

This gives us a center of (58.729, 108.373)

## Force of Gravity

dduck = dPLA \* dInfill;  
duckmass = areaduck \* dduck;  
duckmasswet = areaduckwet \* dduck  
duckforce = duckmass/1000 \* g; %force = M(kg)\*g(N)

Duck Force in 3.59 (mN)

## Submerged Area Values.

To find the submerged area, I used logic matrices and if statements to check if any of a triangles vertex are submerged. Because of the very small triangles, I don’t look for the point of intersection, or create a second shape and re-triangulate. I take a sum of all the triangles that have any vertices below the waterline. This does lead to a slightly inaccurate measurement, as there isn’t a clean waterline, but a jagged approximation that follows the counters of the triangle.

iswet = dely < waterLine; if (v1y < waterLine) && (v2y < waterLine) && (v3y < waterLine)   
 iswetcheck(i) = 0;  
 else  
 iswetcheck(i) = 1;  
 end  
 if (iswetcheck(i) == 1)   
 CenterPerTrianglewet(n,:) = [((v1x + v2x + v3x) / 3), ((v1y + v2y + v3y)/ 3)];  
 plot(CenterPerTrianglewet(n,1),CenterPerTrianglewet(n,2),'g.')  
 n = n+1;  
  
 AreaPerTrianglewet(i) = abs(iswetcheck(i)\*((v1x\*(v2y - v3y)) + (v2x\*(v3y - v1y)) + (v3x\*(v1y - v2y))) / 2);  
 end

areaduckwet is 203.2mm^2

Using the same methods shown above, we also calculate the Buoyant force 1.9914 (mN) and wet center of mass: (60.94, 116.112)

## Question 1: Will the duck float in the illustrated configuration.

No. In order for an item to float, the duck must have a center of mass and a center of buoyancy that are equal and equivalent vectors. In this not case, the center of buoyancy and the center of mass are not in line. This means the duck will not float in the current configuration.

## Question 2 & Question 3 Can we change density or emerged height to show that 2d shape will float in upright configuration. What are the new densities and heights? Otherwise, why does it not float as intended?

No. In order for an item to float, the duck must have a center of mass and a center of buoyancy that are equal and equivalent vectors. As shown by this model, the duck's center of mass and center of buoyancy aren't aligned. The duck will always tip. A new infill density of 27% will establish the waterline at the current a value.

