Magic Wine Bottle Holder Design
1-Bottle and 2-Bottle

JoshTheEngineer

July 17, 2016

Contents

List of Figures

1 Overview

This document is intended to provide some insight into the design process for both the single and double wine bottle holders. The introduction material for both methods is generally the same, while the computation iterations is where the two designs deviate. These deviations will be made clear in the subsequent sections.

I have created three separate ways that you can begin designing. I will discuss each of these in separate sections later in the document.

- MATLAB Code
- C++ Code
- Excel File

The MATLAB code gives you most intuitive control over the design, mainly because it is laid out in an easy-to-use GUI. It also has an immediate visual output, allowing you to see what your design will look like before you even start cutting. Another benefit is that you can see what effect the hole diameter has on how your bottle will lie. The final dimensions are output to the appropriate text box in the GUI.

The C++ file is meant for more advanced users who are already familiar with how command line inputs work. It is a simple, quick way to change design parameters and get immediate output to the command window, or terminal. There are separate files for both the single-bottle and double-bottle designer. You will need to compile and run each separately. It is important to note that if you are using an input file, they are also different for single-bottle and double-bottle.

The Excel file is meant for users that don't have access to MATLAB and don't know how to, or don't want to, run the C++ code. The design is a little more cluttered such that it fits in a single laptop screen while still being readable. The inputs and outputs are clearly marked by their respective colors. The Excel file contains the designer for both single bottle and double bottle designs, albeit on separate worksheets. There are also worksheets that show what the output dimensions are.

The variable names I will be using in this document are the same (as far as I can remember) as those that appear in the code. This allows you to look back and forth between this document and the code, and understand what is happening in each section.

2 Wine Bottle Parameters

There are multiple wine bottle types out in the world. The two most well-known types are the Bordeaux and the Burgundy. The dimensions of typical bottles for these two types are hard-coded into my codes. If you want to enter in a more obscure style of bottle, you can certainly do that as well, using the information presented in this section regarding specification of dimensions.

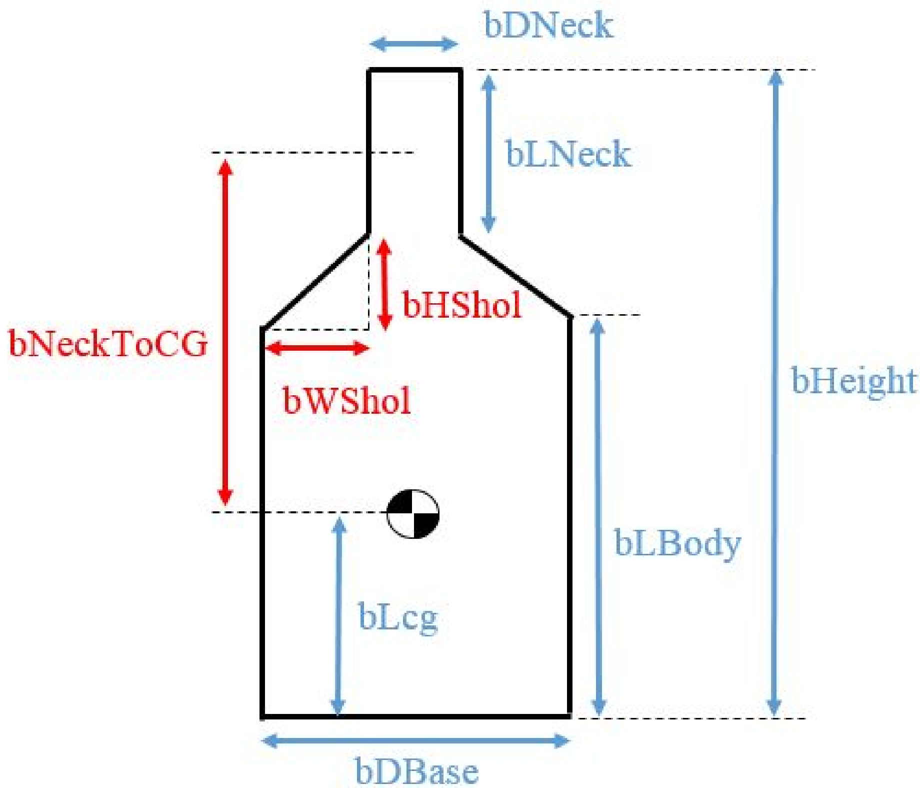

Figure 1: Wine bottle schematic with code variables.

The variables for a typical wine bottle can be seen in Fig. 1. The blue dimensions are those that are supplied by the user. The red variables are calculated in the code. The lowercase b that precedes all the variables indicates that it is a *bottle* variable. You will see in the next section that a lowercase w is used for the *wood*. For the double-bottle holder, there are, of course, two bottles. You can enter in different bottle parameters for the top and bottom bottles. The variables for the bottles will either have a 1 or a 2 after the b to signify the top and bottom bottle, respectively (e.g. $b1LBody$, $b2Height$).

The circle in the middle of the bottle with alternating black and white quarters is the center of gravity (CG) of the bottle when it is lying flat, or horizontal. To find where it is located for an arbitrary bottle, you can lie the bottle down on something cylindrical, like a pen. Roll the bottle back and forth, and the point at which the bottle is balanced is the lateral CG of the horizontal bottle.

The code assumes that this is always the CG of the bottle. This is not always the case, specifically when the bottle does not lie flat in the hole. There is a little air in the bottle, and the wine is allowed to slosh around depending on the bottle orientation. Depending on the size of the hole for the bottle neck, the bottle may lie inclined at an angle, which will move the CG closer to the pivot point, decreasing the moment from the bottle. This will be discussed in a later section.

The red variables can be solved using the existing blue variables in the schematic. The shoulder width, $bWShol$, can be calculated using the base diameter, $bDBase$, and the neck diameter, $bDNeck$, as seen in Eq. (1).

$$bWShol = \frac{bDBase - bDNeck}{2} \tag{1}$$

The shoulder height, $bHShol$, can be calculated using the total height of the bottle, $bHeight$, the length of the body, $bLBody$, and the neck length, $bLNeck$, as seen in Eq. (2).

$$bHShol = bHeight - bLBody - bLNeck \tag{2}$$

Finally, the distance from the neck half-length to the CG can be calculated as seen in Eq. (3). The neck half-distance is used because we are assuming that this point will always be located at the wood half-thickness. This looks aesthetically normal.

$$bNeckToCG = bHeight - bLcg - \frac{bLNeck}{2} \tag{3}$$

3 Wood Parameters

There are fewer wood parameters than bottle parameters. One reason is that we are assuming a rectangular block of wood, so to define its dimensions we only need the length (wL), width (wW), and thickness (wT).

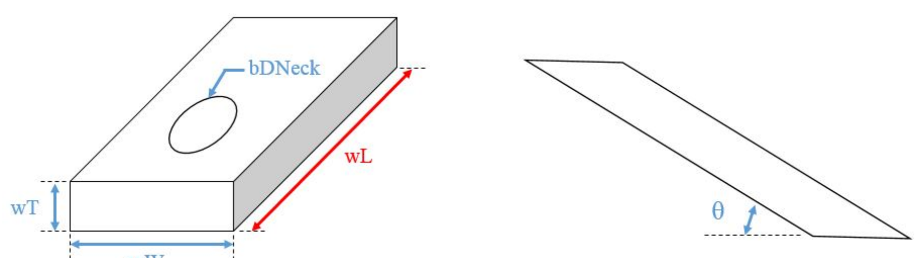

Figure 2: Wood schematic with code variables.

In Fig. 2, both the width and the thickness are shown in blue, because these are set by the user and stay constant throughout the computations. The length is shown in red, because this is one of the main design variables that is being computed during the iterations. The angle θ is a user-defined design parameter, and is the angle the wood board makes with the table. In the code, another angle, δ, is defined as $90 - \theta$ (in degrees).

4 Minimum Angle Calculation

There are two ways to define the angle in the code. The first is to specify an arbitrary angle, while the second is to specify that you want to design for the minimum angle. The minimum angle is the angle at which the shoulder of the wine bottle makes contact with the wood board. Either way, the minimum angle needs to be calculated, if only to ensure that the user-defined angle is not less than the minimum allowable angle, resulting in an impossible solution.

Two iterations are needed to adequately solve for the minimum angle. If we first assume the wood board is infinitely thin, then the bottle shoulder will contact the wood when it rests at some angle, as seen in Fig. 3 as the black line (passing through the neck half-length and touching the shoulder). We can define a minimum angle based off this infinitely thin piece of wood, but we know the wood has some thickness, as it is specified at the beginning of the calculations. This will push the minimum angle to a larger value than that for the thin wood. We will need the thickness parallel to the table for the calculation, but that depends on the angle of the wood with respect to the table, which we don't know yet. To start, we can use the wood half-thickness, which is seen in the right in Fig. 3.

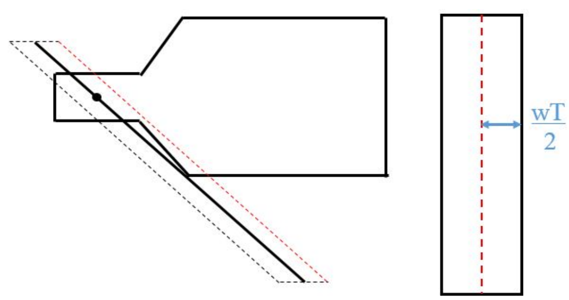

Figure 3: Minimum angle: wood board with no thickness.

We can solve for the initial minimum angle by looking at Fig. 4. The dashed red line is the center-line of the wood board. This center-line is offset from the shoulder by our initial guess of the thickness (the half-thickness of the wood). The quantities shown in blue are known, while those in red need to be computed.

To solve for θ, we need the variables y and x, which are solved for in Eqs. (4) and (5), respectively.

$$y = \frac{bDBase}{2} \tag{4}$$

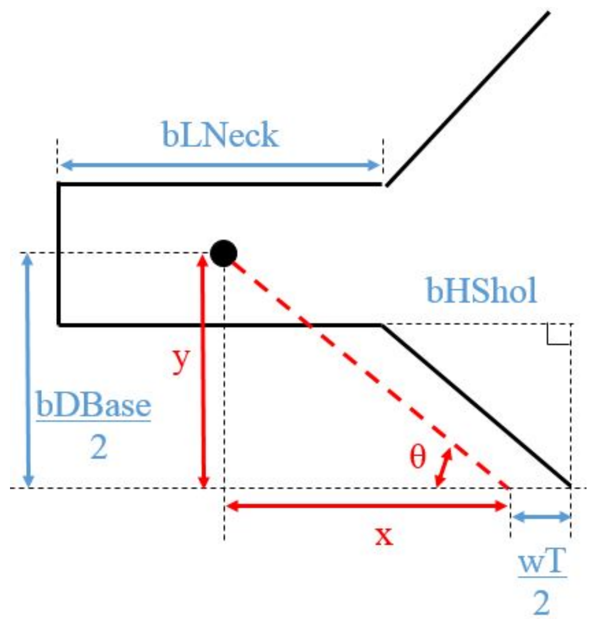

Figure 4: Schematic to solve for minimum angle using wood half-thickness.

$$x = \frac{bLNeck}{2} + bHShol - \frac{wT}{2} \tag{5}$$

The initial minimum angle can then be calculated as seen in Eq. (6).

$$\theta = \tan^{-1}\left[\frac{y}{x}\right] \tag{6}$$

We know this is not the actual minimum angle, because we have not yet accounted for the thickness of the wood parallel to the table. Using the angle we computed above, we can define the half-thickness that we need to account for, as seen in Fig. 5. The wood half-thickness and initial angle are known, so

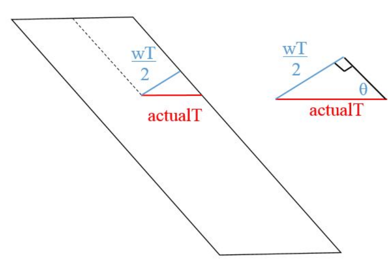

Figure 5: Schematic to solve for actual thickness of wood.

we can solve for the actual thickness, $actualT$, in Eqs. (7) and (8).

$$\sin(\theta) = \frac{(\frac{wT}{2})}{actualT} \tag{7}$$

$$actualT = \frac{(\frac{wT}{2})}{\sin(\theta)} \tag{8}$$

Now we can update the picture used to calculate the minimum angle accounting for this new thickness, seen in Fig. 6. The variable y has not changed, and is still given in Eq. (4). The new equation for x is given in Eq. (9). The actual minimum angle can then be solved for as before, in Eq. (6).

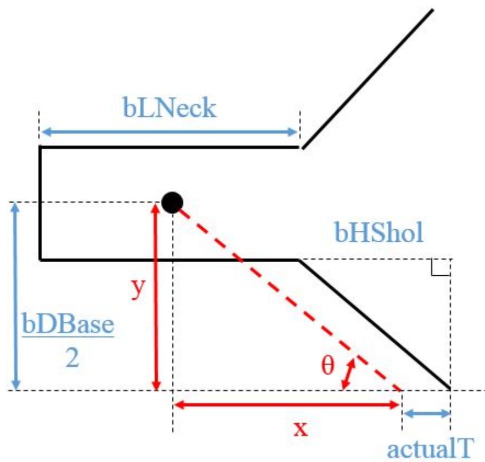

Figure 6: Schematic to solve for minimum angle using actual thickness of wood.

$$x = \frac{bLNeck}{2} + bHShol - actualT \tag{9}$$

It's clear from the preceding discussion that this new minimum angle will inevitably lead to a new actual thickness, which then leads to a new minimum angle, and so forth. However, it seems that these two iterations are enough to get close enough to the actual minimum angle, and further iterations are redundant. If the user chooses to use the minimum angle for the design, then this angle is used. If the user does not want to use the minimum angle for the design, and instead wants to use their own desired input, then their input is compared against the minimum angle to ensure it is larger.

5 Computation Iterations

In the subsections below, we will go through the iterative loop used for designing for the correct bottle placement. The process up until this point has been the same for both the single-bottle and the double-bottle designs. Here, however, we will go over each method separately as they differ slightly. For both designs, the user chooses a maximum number of iterations to allow, and the code loops over that specified number. If the design converges before the number of maximum iterations is reached, the code stops and displays the results. Otherwise, the code will reach the maximum iterations and notify the user that the design did not converge. The maximum number of iterations allowed will probably need to be increased if this happens. The number of iterations required for the double-bottle design is greater than that required for the single-bottle design.

5.1 Single-Bottle

The following describes the computations for a single iteration. It starts by setting a variable called *bLPivotOld* equal to $bLPivot$, which is initially set to zero to start off the calculations. By setting its value equal to zero, it's saying that the bottle CG is located vertically over the base of the wood (i.e. where the wood touches the table). See Fig. 7 for a simplified schematic. This design

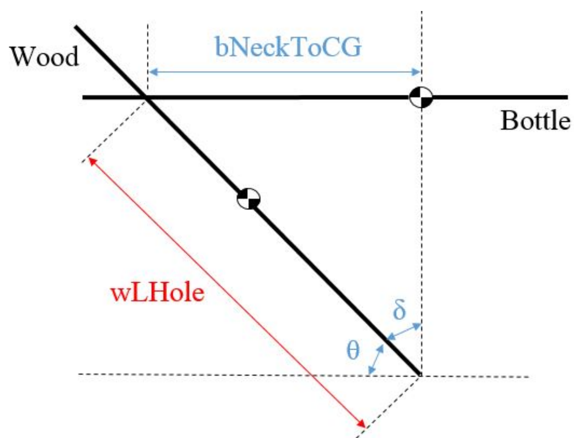

Figure 7: Initial CG locations for first iteration.

obviously won't work, because while the bottle CG is over the base, the wood CG is not, and will contribute to a destabilizing moment. The total system CG will then not be located at the base, the system will tip over, and you'll have broken shards of glass lying in puddles of wine on your table. This is why we need to iterate.

The next step is to find the length of the center of the hole from the base of the wood, $wLHole$, using Eq. (33).

$$wLHole = \frac{bNeckToCG - bLPivot}{\sin(\delta)} \tag{10}$$

For the first iteration, *bLPivot* is zero, but as the iterations progress, this value changes to positive values as the bottle is shifted. An updated schematic can be seen in Fig. 8. Note that we are still using Eq. (33), but now the top of the triangle (opposite δ) is no longer just $bNeckToCG$, it is $bNeckToCG - bLPivot$. So how did we figure out what this new value of $bLPivot$ was? Let's move on through the iteration to figure that out.

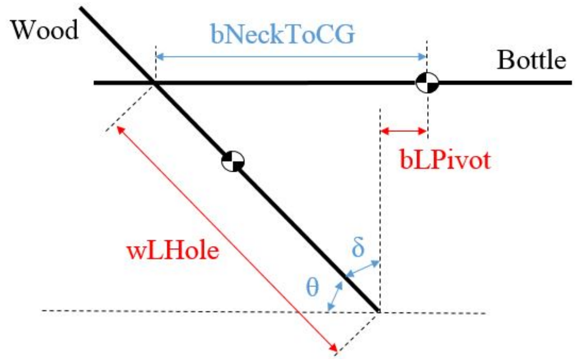

Figure 8: Subsequent CG locations for for the rest of the iterations.

Once we have the value of $wLHole$, we need to find the length of the wood associated with that value by taking into account the bottle neck diameter and the aesthetic parameter, $pctLAdd$. By looking at Fig. 9, we can find the values of $wL1$ and $wL2$, in Eqs. (11) and (12), respectively.

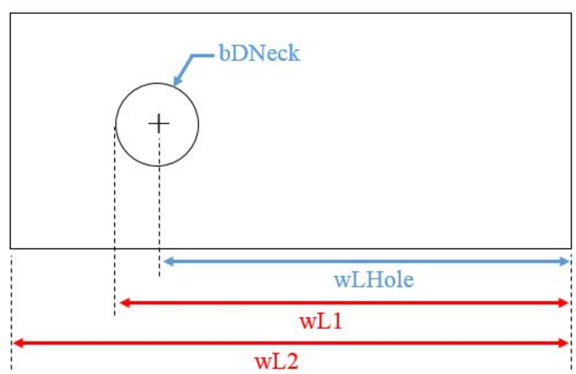

Figure 9: Wood length definitions for single-bottle design.

$$wL1 = wLHole + \frac{bDNeck}{2} \tag{11}$$

$$wL2 = wL1 + (\frac{pctLAdd}{100})wL1 \tag{12}$$

We can see that Eq. (11) added half of the bottle neck diameter to the length of the hole. This effectively encloses the bottle in the wood, and would technically work for this design. However, it looks quite weird and might not be structurally sound since the wood holding the bottle neck in is very thin. This is where the aesthetic parameter comes into play. From Eq. (12), we are adding a percentage of the "must-have" wood, and end up with something that looks normal.

We are not quite done, as we still need to account for the length that we are losing when we cut the angles at the top and bottom. Take a look at Fig. 10. We know the length of the wood that we approximated as a line ($wL2$) when

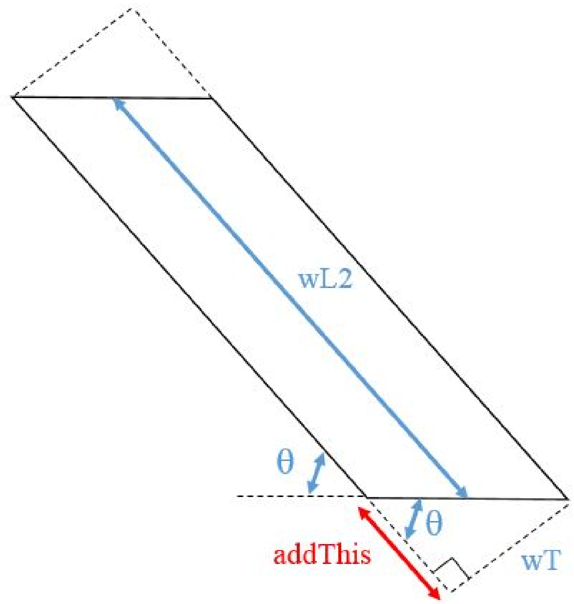

Figure 10: Additional wood length added due to non-zero thickness.

solving for the CG locations. The wood has a thickness wT, and when we cut the angles off the ends of the wood, we need to make sure that the length $wL2$ is still retained at the center-line. If we look at the blue center-line, we can see that we will need to extend out the top a little, and also extend out the bottom a little. Another way of looking at this is to shift the blue center-line over to the left edge of the wood, and notice that now we only need to extend out the bottom a distance of $addThis$, which is solely defined by θ and wT, which are both known and don't change throughout the program. The distance $addThis$ can be solved for using Eqs. (13) and (14).

$$\tan(\theta) = \frac{wT}{addThis} \tag{13}$$

$$addThis = \frac{wT}{\tan(\theta)} \tag{14}$$

The final length of the wood for the current iteration is seen in Eq. (15), and from this we will be able to calculated the wood's mass to use in our moment calculations.

$$wL = wL2 + addThis \tag{15}$$

Now that we have the wood length, let's calculate the wood's weight. The volume, mass, and weight can be computed in Eqs. (16)-(18).

$$wVolume = (wT)(wW)(wL) \tag{16}$$

$$wMass = (wRho)(wVolume) \tag{17}$$

$$wWeight = (wMass)(9.81\frac{m}{s^2}) \tag{18}$$

The volume of the wood is in $[m^3]$, the mass in $[kg]$, and the weight in $[N]$. Now we need to solve for the horizontal length from the base of the wood to the wood's CG, $wLPivot$. The assumption being made is that the wood is a single rectangular piece of wood, without a hole. The implication of this assumption is that the CG of the wood is, in reality, farther left (when looking at our schematics) than it is in our computations. However, the effect is so small that we can safely ignore it. Using this assumption, we can say that the wood's CG is located at its halfway point, or $\frac{wL}{2}$. From Fig. 11, we can find the value of $wLPivot$ using Eq. (40).

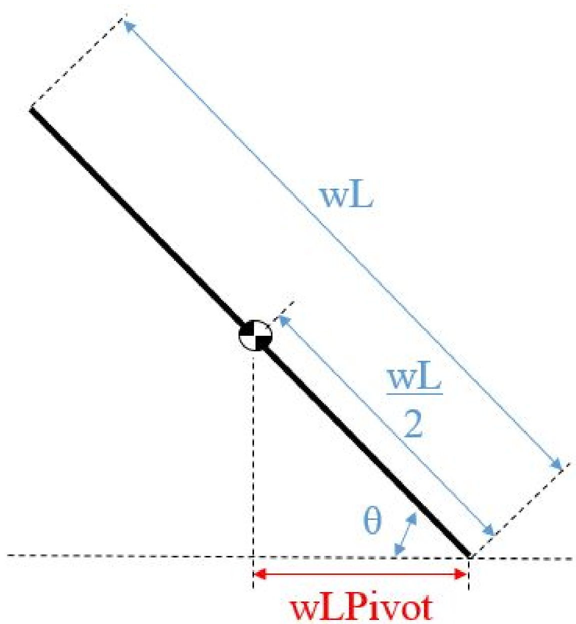

Figure 11: Horizontal pivot location of wood CG.

$$wLPivot = (\frac{wL}{2})\cos(\theta) \tag{19}$$

Now we can calculate where the bottle CG needs to be located (parallel to the table) such that the system is stable. We will use a moment balance. The moment about the base from the wood CG and the bottle CG should be zero. By looking at Fig. 12, we can write Eq. (20). The signs of the terms follows the right-hand-rule; the wood creates a positive moment about the base, while the bottle produces a negative moment.

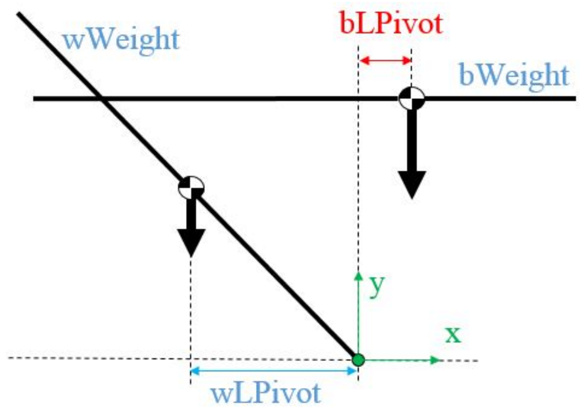

Figure 12: Variables needed for moment calculation.

$$(wWeight)(wLPivot) - (bWeight)(bLPivot) = 0 \tag{20}$$

We can rearrange Eq. (20) to solve for the unknown, $bLPivot$, as seen in Eq. (21).

$$bLPivot = \frac{(wWeight)(wLPivot)}{bWeight} \tag{21}$$

Now we can compare this new bottle pivot length to the old one that we set at the beginning of the iteration, and if the difference in their values is still above a certain error tolerance, we continue to iterate. If their difference is below the error tolerance, then we can break out of the iterative cycle, and we have our final design! As a final check, we need to make sure that the final design does not force our bottle to touch the table. This is called the *clearance*, and can be seen in Eq. (22) by referencing Fig. 13.

$$clearance = (wLHole)\sin(\theta) - \frac{bDBase}{2} \tag{22}$$

Now the only thing left to do is figure out what diameter hole we need for the bottle neck. Skip to Section 6 for that discussion.

5.2 Double-Bottle

The method of solution for the double-bottle design is slightly different than that for the single-bottle design, and will be discussed in this section. Before we move into the iterative process, there is a variable called *minSep* that needs to be solved for first. The user sets a variable called *sepPct*, which dictates how

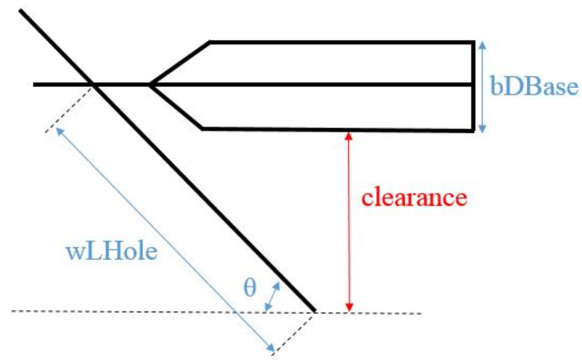

Figure 13: Clearance between bottle and table.

far (vertically) the bottles' neck holes will be separated from each other. This is merely an aesthetic parameter (similar to *pctLAdd*), but obviously affects the overall design. The input *sepPct* is a percentage, as the name suggests, so first we divide it by 100 to get it into decimal form. In the variable names, "1" refers to the bottom bottle and "2" refers to the top bottle. The value of *minSep* is calculated using Eqs. (23) and (24).

$$minSep1 = (sepPct)(b1DBase) + (sepPct)(b2DBase) \tag{23}$$

$$minSep = minSep1 + (sepPct)(minSep1) \tag{24}$$

Take a separation percentage of 50, for example. From Eq. (23), the vertical separation of the holes will be equal to the sum of each of the bottle base radii. This means that the bottles will just be touching. Equation (24) then adds that vertical height to 50 percent of that height to get the final vertical separation. This minimum separation can be defined differently, but I have found that this works well. The *minSep* value does not change after it is initially calculated, because it only depends on the bottle diameters. This means that the bottle separation will remain fixed throughout the iterations, and the bottles will essentially shift up/down as a unit during the proceeding calculations.

To get ready for the iteration (and to not over-calculate values that won't change), we can solve for the X-distance separation of the two bottle CGs by looking at Fig. 14. If the two bottles were identical (which is usually the case), then we can solve for CG X-distance separation using Eq. (25).

$$xSepCG = \frac{minSep}{\tan(\theta)} \tag{25}$$

However, when the two bottles have different geometries, this equation no longer holds, and we must use the following technique. First, we solve for the X-distance separation of the neck half-lengths, seen in Eq. (26). Another way of saying this is that we are solving for the X-distance separation of the bottle

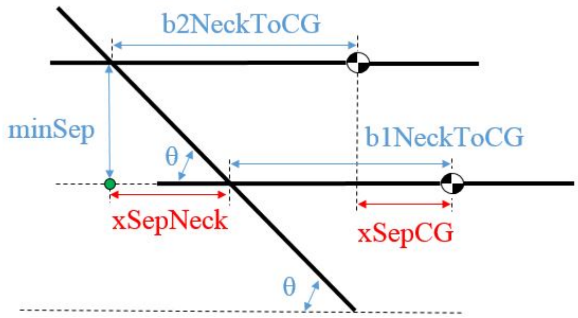

Figure 14: Horizontal separation of bottle CG positions.

neck holes.

$$xSepNeck = \frac{minSep}{\tan(\theta)} \tag{26}$$

We want the X-distance from the green dot in Fig. 14 to the CG point of each bottle, which we will call $x1CG$ for the bottom bottle and $x2CG$ for the top bottle (check Fig. 1 as well). We can see the result of these calculations in Eqs. (27) and (28).

$$x1CG = xSepNeck + b1NeckToCG \tag{27}$$

$$x2CG = b2NeckToCG \tag{28}$$

Now, to find the X-distance separation of the two CGs, we use Eq. (29).

$$xSepCG = x1CG - x2CG \tag{29}$$

To start the iterations, we need to set the initial position of the bottle pivot locations, $b1LPivot$ and $b2LPivot$ for the bottom and top bottle, respectively. We will set the bottom bottle's CG directly over the base (equal to zero, since that's where we define our origin), and the top bottle's CG location follows from their CG separation distance we just solved for, as seen in Eq. (30).

$$b2LPivot = b1LPivot - xSepCG \tag{30}$$

There are just two more variables we set before iteration commences. The first is a moment-adjustment factor, called $adjustM$ (set to 0.01), and is used to move the bottle system left or right. The other variable is called $oldM$ (set to zero), and is how we will see whether our solution has converged. We will see these in action later.

Finally, it's time to enter the iterative loop! We can solve for the lengths from the base to the bottom and top holes by looking at Fig. 15. In Eq. (31), we are solving for the distance from the base to the bottom hole. In Eq. (32), we solve for the distance from the bottom hole to the top hole. In Eq. (33), we solve for the distance from the base to the top hole.

$$w1LHole = \frac{b1NeckToCG - b1LPivot}{\sin(\delta)} \tag{31}$$

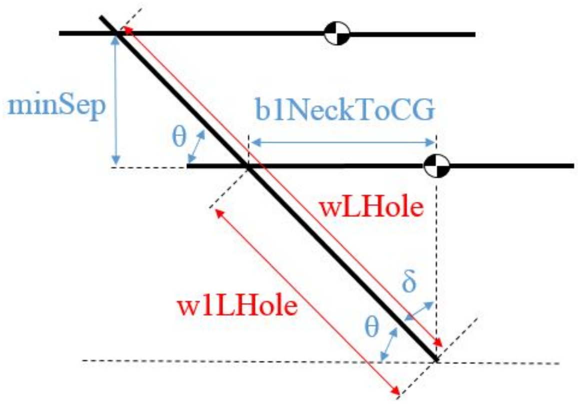

Figure 15: First iteration CG positions.

$$w2LHole = \frac{minSep}{\sin(\theta)} \tag{32}$$

$$wLHole = w1LHole + w2LHole \tag{33}$$

Now we can calculate the length of the wood by referring to Fig. 16. We compute $wLTot1$ by taking into account the top bottle's neck diameter as seen in Eq. (34). Then we compute $wLTot2$ by using $pctLAdd$ as seen in Eq. (35). Finally, we need to add a certain amount of length to take into account the angled cutting, similar to the single-bottle holder (refer to Fig. 10), seen in Eq. (36).

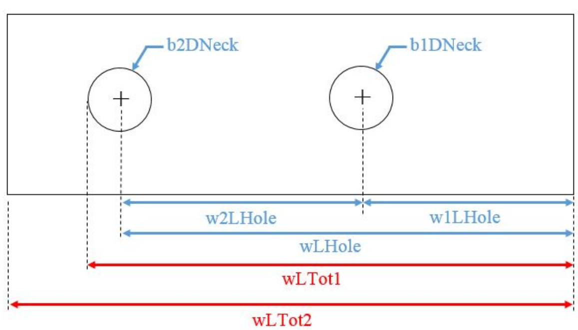

Figure 16: Wood length parameters for double-bottle design.

$$wLTot1 = wLHole + \frac{b2DNeck}{2} \tag{34}$$

$$wLTot2 = wLTot1 + \frac{pctLAdd}{100}(wLTot1) \tag{35}$$

$$wLTot = wLTot2 + \frac{wT}{\tan(\theta)} \tag{36}$$

Now that we have the length of the wood for this iteration, we can calculate the wood's volume $[m^3]$, mass $[kg]$, and weight $[N]$ as seen in Eqs. (37), (38), and (39), respectively.

$$wVolume = (wT)(wW)(wLTot) \tag{37}$$

$$wMass = (wRho)(wVolume) \tag{38}$$

$$wWeight = (wMass)(9.81\frac{m}{s^2}) \tag{39}$$

Next, we need to find the X-distance from the base to the wood's CG. Referencing Fig. 17, we can use Eq. (40) to compute this value. The negative sign is included because the wood's pivot location is always going to be to the left of the origin.

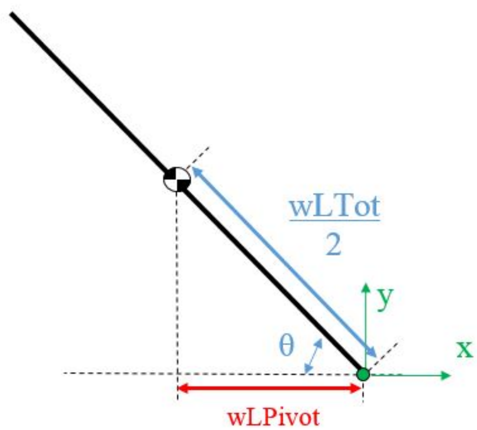

Figure 17: CG position of the wood.

$$wLPivot = -(\frac{wLTot}{2})\cos(\theta) \tag{40}$$

Using the weights of the bottles and wood, along with the pivot distances, we can compute the sum of the moments, $sumM$. When the sum of the moments about the base is equal to zero, then we have a balanced system. Equation (41) is used to compute the moment.

$$\begin{split} sumM = -(wWeight)(wLPivot) - (b2Weight)(b2LPivot) \\ - (b1Weight)(b1LPivot) \end{split} \tag{41}$$

You'll note the minus signs in front of each moment term in Eq. (41). This makes sure that the sign of the resulting moment is in line with the right-hand-rule. Based off the axes shown in Fig. 17, a moment that tips the system to the left will be positive, while a moment that tips the system to the right will be negative. The value of $wLPivot$ will always be negative, and will thus create a positive moment about the base. The double negative in Eq. (41) ensures this. The bottom bottle's CG will always be to the right of the base, and will thus always be positive. To make sure that the bottom bottle creates a negative moment, the minus sign is needed in front of its term.

Now that we have the moment of the system, we first check that the moment has not changed sign from the old stored moment, $oldM$. Recall that the initial value of $oldM$ is zero. To check if the moment value has changed sign, we multiply $oldM$ by $sumM$, and check if the resulting value is negative. If it is, then we refine the moment adjustment parameter, $adjustM$. What are we doing with this adjustment parameter?

First we check that the moment hasn't converged below a certain tolerance, $errorTol$. If it has, then the loop breaks, and the design is finalized. If not, then we need some way to adjust the bottle placement such that we can converge on the correct zero moment, and thus the correct design. The $adjustM$ parameter moves the bottom bottle's pivot either left or right, depending on the sign of the calculated moment, $sumM$. If $sumM < 0$, then the system will tip to the right, and the bottom bottle's pivot location is updated as seen in Eq. (42).

$$b1LPivot = b1LPivot - adjustM \tag{42}$$

If $sumM > 0$, then the system will tip to the left, and the bottom bottle's pivot location is updated as seen in Eq. (43).

$$b1LPivot = b1LPivot + adjustM \tag{43}$$

The new pivot length of the top bottle can be computed as we did before in Eq. (30).

The program iterates until the moment has converged below some error tolerance. Once this happens, the clearance is computed using Eq. (44) (refer to Fig. (13) for a closely related figure).

$$clearance = (w1LHole)\sin(\theta) - \frac{b1DBase}{2} \tag{44}$$

Now it's time to talk about the hole diameters in Section 6 below.

6 Hole Sizing

Now that the wood sizing has been defined, it is time to determine the diameter of the hole we need to drill. There are two limits on the hole size. The absolute minimum diameter hole we can use is the diameter of the bottle neck (and even this is actually smaller than what we would physically be able to use). The absolute maximum diameter hole we can use it technically the width of the wood board. It clearly doesn't make sense to use this as a realistic maximum diameter. Instead, we define the maximum hole diameter as the diameter that would allow our bottle to lie flat (horizontal with respect to the table). It is important to note that all the holes are assumed to be drilled perpendicular to the wood. This is mainly done because drilling angled holes is considerably more difficult if all you have is a hand drill. You can most certainly drill an angled hole if you so desire, but those dimensions are not considered here. I have also found that perpendicular holes look just fine as far as aesthetics are concerned.

The hole diameter can then clearly range anywhere between the minimum and maximum diameters. When the diameter is at its minimum, the bottle will lie perpendicular to the board. When the diameter is set to its maximum, the bottle will lie parallel to the table. As we increase the hole size from the minimum to the maximum, the angle of the bottle will change, and become more and more parallel to the table. The minimum hole diameter is defined in Eq. (45).

$$minHoleD = bDNeck \tag{45}$$

The maximum hole diameter (also called the flat hole diameter), can be found using Eqs. (46) - (48) by looking at Fig. 18. The variable a is governed solely by the contribution from the bottle neck diameter, while the variable b is governed solely by the contribution of the thickness of the wood.

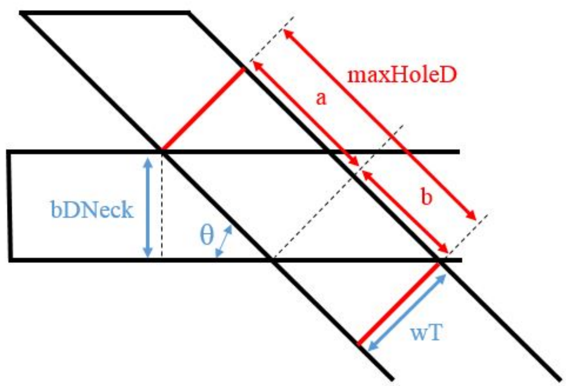

Figure 18: Maximum hole diameter schematic.

$$a = \frac{bDNeck}{\sin(\theta)} \tag{46}$$

$$b = \frac{wT}{\tan(\theta)} \tag{47}$$

$$maxHoleD = a + b \tag{48}$$

As mentioned previously, any hole diameter between the minimum and maximum is allowed, but they will result in different bottle angles with respect to the table. For example, if we want the wood to be angled 45^o, and we want the minimum diameter hole, then the bottle will lie actually lie at an angle of 135^o from the horizontal. The way we get this angle is to imagine the wine bottle sitting normally on the table. Now hold the bottle at the neck and rotate the bottle 135^o. This is the angle it will be resting at in the MWBH. For the bottle to be lying horizontal to the table, as it would be for the case of the maximum hole diameter, we would need to rotate the bottle 90^o.

For any arbitrary hole diameter between the minimum and maximum, we can solve for this angle by referencing Fig. 19. The black lines outline *just* the hole in the wood, while the green lines are the bottle neck. We are looking for

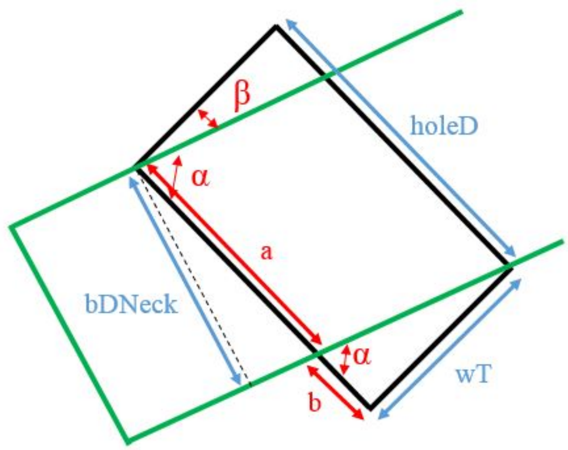

Figure 19: Bottle angle from hole diameter.

the angle β. This angle will be equal to zero when the hole diameter is at a minimum (the diameter of the bottle neck). You can imagine this by shrinking the hole diameter, and seeing that physically the bottle has to rotate farther and farther counter-clockwise. Similarly, if the hole diameter increases, the bottle will rotate clockwise. Since we are looking for β, we can just as easily look for α, and convert to β after, since $\beta = 90 - \alpha$. The key is then to find an equation for the unknown α that is solely a function of the known variables (in blue).

Let's first find relations for the variables a and b, shown in Eqs. (49) and (50).

$$a = \frac{bDNeck}{\sin(\alpha)} \tag{49}$$

$$b = \frac{wT}{\tan(\alpha)} \tag{50}$$

Now we can write Eq. (51) by summing these two.

$$holeD = a + b = \frac{bDNeck}{\sin(\alpha)} + \frac{wT}{\tan(\alpha)} \tag{51}$$

This can also be written as Eq. (52).

$$holeD = \frac{bDNeck}{\sin(\alpha)} + \frac{\cos(\alpha)wT}{\sin(\alpha)} \tag{52}$$

Now multiply both sides by $\sin(\alpha)$ to get Eq. (53), and then bring all the terms to one side in Eq. (54).

$$\sin(\alpha)holeD = bDNeck + \cos(\alpha)wT \tag{53}$$

$$\sin(\alpha)holeD - \cos(\alpha)wT - bDNeck = 0 \tag{54}$$

Using Eq. (54), we can use a root finding method to find the value of α that satisfies the expression. As mentioned previously, the value of β can be found by subtracting α from 90. The angle of the bottle with respect to the table (using the rotation method mentioned before) is given in Eq. (55).

$$rotAng = 180 - \theta - \beta \tag{55}$$

This is the angle that the MATLAB code uses to plot the bottle because of the way the bottle is initially plotted before rotations are applied. If we want the angle of the bottle relative to the table, which is more useful to the designer, then we can use Eq. (56).

$$bottleAngle = \theta - \beta \tag{56}$$

When the $bottleAngle = 0^o$, the bottle is lying horizontal with respect to the table, while when the $bottleAngle = 45^o$, the bottle is lying upwards at an angle of 45^o with respect to the table.

7 Discussion of Assumptions

Coming soon!

8 Design: MATLAB

There are many similarities between the single and double bottle designs. I'll describe all the overlapping topics here, and then breaking down the differences in the sections below.

8.1 Single-Bottle

Coming soon!

8.2 Double-Bottle

Coming soon!

9 Design: C++

Coming soon!

9.1 Single-Bottle Code

Coming soon!

9.2 Double-Bottle Code

Coming soon!

9.3 Single-Bottle Input File

Coming soon!

9.4 Double-Bottle Input File

Coming soon!

10 Design: Excel

Coming soon!

10.1 Single-Bottle

Coming soon!

10.2 Double-Bottle

Coming soon!

10.3 Adding Woods

Coming soon!