

Objective

This project is based on the *Grand Challenge HC18 Automated measurement of fetal head circumference* (Thomas 1). Specifically, this project is focused on developing a process to automatically measure fetal head circumference when provided with a standard ultrasound image. The challenge was formed due to the importance of fetal head circumference measurements:

“Fetal HC biometric measurement is an important item of fetal examination, which is generally measured in the specific cross section of fetal head (called standard plane). Obstetricians and gynecologists can estimate the gestational age and fetal weight at the 13th to 25th week of pregnancy by measuring the fetal HC, evaluate the development of the fetus, and determine the delivery mode of pregnant women [3]. At present, the fetal HC is generally measured manually by radiologists with ellipse fitting, which is time-consuming and tedious, and may cause inter- and intra-operator difference. Therefore, there is a need for developing computerized automatic fetal HC biometric measurement methods.” (Zeng et al)

Additionally, my approach to implementing head circumference measurement was performed without the use of a neural network.

Data Set

The HC18 provided training and test data sets. ‘The data is divided into a training set of 999 images and a test set of 335 images. The size of each 2D ultrasound image is 800 by 540 pixels with a pixel size ranging from 0.052 to 0.326 mm.’ (Thomas 1) A csv file was provided with the training data with the file name, pixel size, and the actual head circumference (mm). The pixel size allows the user to accurately calculate the circumference, as this will change depending on the depth the ultrasound probe is currently at when the image is captured.

Only the training data was used in this implementation as there is no intention of submitting to the challenge itself. There were some duplicate images in the training set. These were not used. The total number of images used from the training set was 806.

A review of the data sets provided contextual information that was useful in designing the implementation. The images are equally split across first, second, and third trimester. The maximum and minimum circumferences are 346.4mm and 44.3 respectively. An inspection of the images also revealed that the quality of the images is not consistent. There was significant variation in the image quality. Some images are clear and in focus whereas others are blurry and off center.

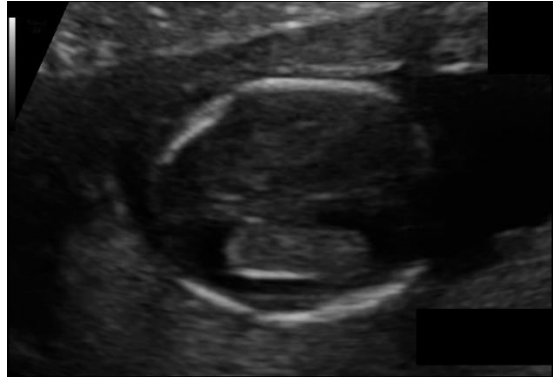


Figure 1: Blurry image with shadowing 147_HC.png

Implementation

This implementation used the following python libraries:

- Numpy
- Cv2
- Matplotlib.pyplot
- Pandas

Google Colab was used for iterating and testing, but the final code submission is via jupyter notebook.

Preprocessing

Once the image was imported into a variable, I ensured the image was in a grayscale format. A median blurring filter of size 5x5 was applied to smooth the image. The median blurring produced better results than the Gaussian filter of the same size.

I then performed a binary threshold on the blurred image, with the mean value of the blurred image as the threshold. I did attempt differing threshold values, mean +/- 5 or 10, 127, etc. , but the mean had the best and most consistent outcomes.

Next, I eroded and dilated the image. The first iteration was with a 5x5 for both the eroding and dilation. With the second iteration, I used a 9x9 filter for the erosion process and the standard 5x5 for the final dilation.

This is then followed by extracting the edges by subtracting the erode/dilated image from the thresholded image.

I wrapped the entire preprocessing effort into a single function for simplification's sake and returns an image that consists solely of edges. There is an optional argument to allow for the before and after images to be shown on each call. By default, it is set to false.

Preprocessing

```

1 def preprocess(image, show=False):
2
3     # median blurring allows for better results
4     blur = cv2.medianBlur(image, 5)
5
6     # find the mean of the image
7     im_mean = np.mean(blur)
8
9     # apply binary threshold on the mean
10    ret,b_img = cv2.threshold(blur, im_mean, 255, cv2.THRESH_BINARY)
11
12    # structure
13    struct = np.ones((5,5), np.uint8)
14    struct1 = np.ones((9,9), np.uint8)
15
16    edi_img = cv2.erode(b_img, struct)
17    edi_img = cv2.dilate(edi_img, struct)
18    edi_img = cv2.erode(edi_img, struct1)
19    edi_img = cv2.dilate(edi_img, struct)
20
21    # subtract the eroded image from the initial image
22    edged_img = b_img - edi_img
23
24    if show == True:
25        show_side_by_side(image, edged_img, title2='Boundary Extraction')
26
27    return edged_img

```

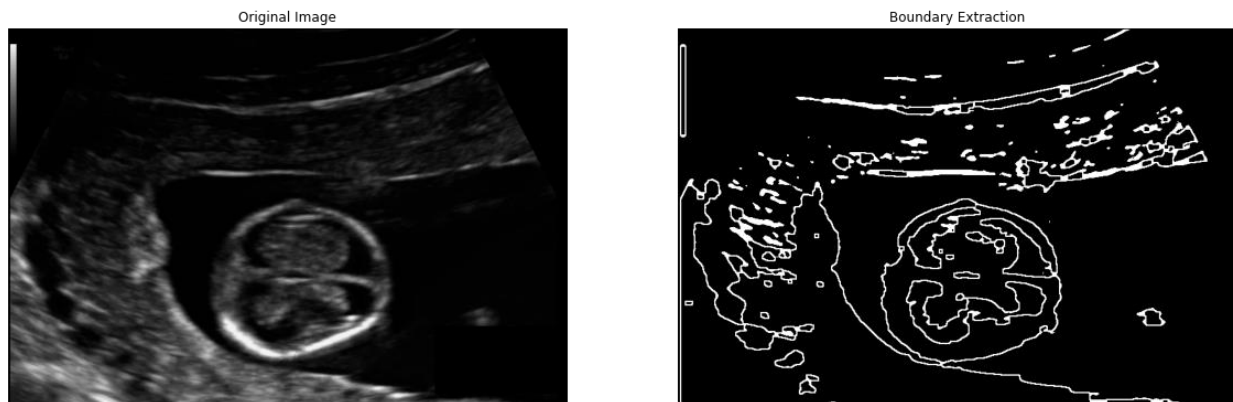


Figure 2: Original Image & Edged Image

Fitting an Ellipse to the head perimeter

Finding the head perimeter proves to be a difficult problem to solve. The current iteration starts by finding the contours. Each contour is then evaluated for fitness as a potential candidate to minimize false positives. This fitness evaluation is a simple check of minimum length and overall area of the contour is above a certain threshold. In this instance the minimum length of the contour is 500 and the minimum area is 25,000.

A convex hull is attempted to be calculated for each contour that is evaluated as a potential fit. This convex hull is performed in a counterclockwise fashion. No, measurable performance changes were noted between clockwise or counterclockwise. An ellipse is fit to each convex hull that is generated.

These ellipses are then checked to ensure they are within the bounds of the image. The check evaluates the position for 8 points on the ellipse, specifically at every 45 degrees. The difficulty with this check is based in the fact that the ellipse is rotated, offset from a center position of (0,0), and the coordinate system between the equation and the image are independent of each other. The equation below allows for the rotated and offset position to be calculated. For each position check, x and y, had to be swapped and compared against the boundaries of the image.

Step 1 - Parametric Equation of an Ellipse

The parametric formula of an Ellipse - at (0, 0) with the Major Axis parallel to X-Axis and Minor Axis parallel to Y-Axis:

$$\begin{aligned}x(\alpha) &= R_x \cos(\alpha) \\y(\alpha) &= R_y \sin(\alpha)\end{aligned}$$

Where:

- R_x is the major radius
- R_y is the minor radius

Step 2 - Rotate the Equation

To rotate any formula we use the rotation mapping:

$$\begin{aligned}x &= t \cos(\theta) - f(t) \sin(\theta) \\y &= t \sin(\theta) + f(t) \cos(\theta)\end{aligned}$$

Where:

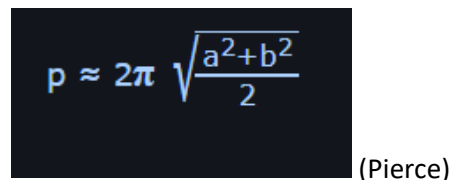
- θ is the rotation angle
- t is the parameter of the original function
- $f(t)$ is the original function

Once we put the Ellipse equation in the rotation equation we get:

$$\begin{aligned}x(\alpha) &= R_x \cos(\alpha) \cos(\theta) - R_y \sin(\alpha) \sin(\theta) \\y(\alpha) &= R_x \cos(\alpha) \sin(\theta) + R_y \sin(\alpha) \cos(\theta)\end{aligned}$$

Figure 3: Rotated Ellipse (Gil 1)

For any ellipse that remained, the perimeter would then be calculated using an approximation. This approximation is within 5% and only holds true if the minor and major do not have a 3x difference between themselves.



$$p \approx 2\pi \sqrt{\frac{a^2+b^2}{2}} \quad (\text{Pierce})$$

All the ellipses and contours are drawn on an image of the same size. The function returns this image, the perimeter, and the dimensions of the ellipse.

Step 3 - Shift the Equation from the center at (0, 0)

To shift any equation from the center we add C_x to the x equation and C_y to the y equation. Therefore the equation of a Rotated Ellipse is:

$$\begin{aligned}x(\alpha) &= R_x \cos(\alpha) \cos(\theta) - R_y \sin(\alpha) \sin(\theta) + C_x \\y(\alpha) &= R_x \cos(\alpha) \sin(\theta) + R_y \sin(\alpha) \cos(\theta) + C_y\end{aligned}$$

Where:

- C_x is center X.
- C_y is center Y.
- R_x is the major radius.
- R_y is the minor radius.
- α is the parameter, which ranges from 0 to 2π radians.
- θ is the Ellipse rotation angle.

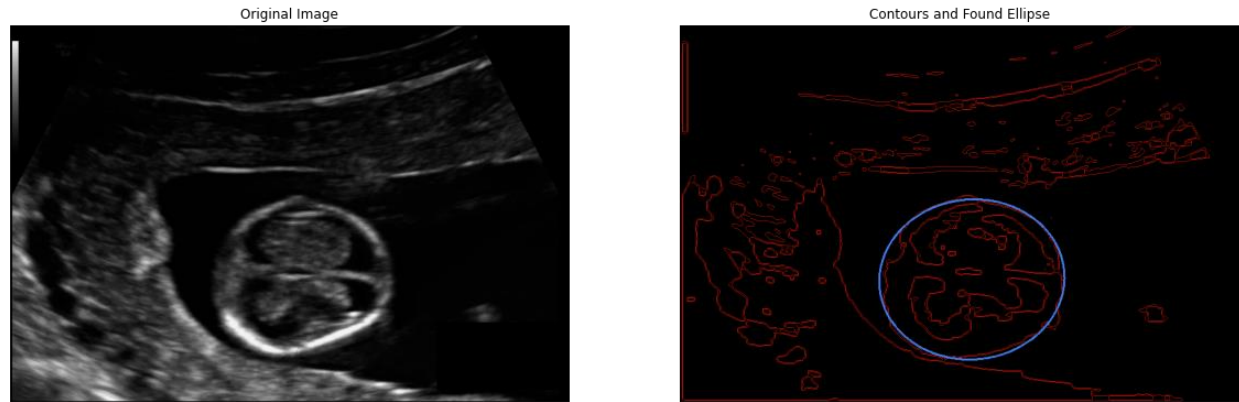


Figure 4: Original Image & Found Ellipse - 003_HC.png

Results

Of the 806 images in the training set, this approach was able to find an ellipse for 205. This is roughly 25.43%. The median absolute deviation was 61.96 with a standard deviation of 81.35.

If we excluded the false positives, we are left with 151 images which is approximately 18.73%. The median absolute deviation drops to 27.51 with a standard deviation of 39.90.

Below is an example of a successful result

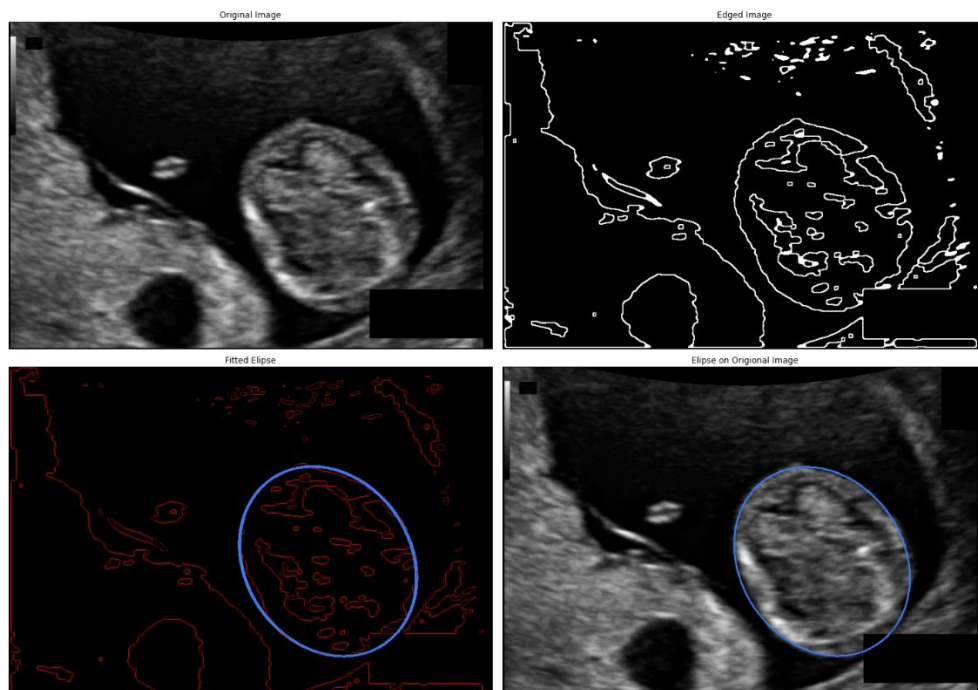


Figure 5: File 025_HC.png

Below is an example of a false positive

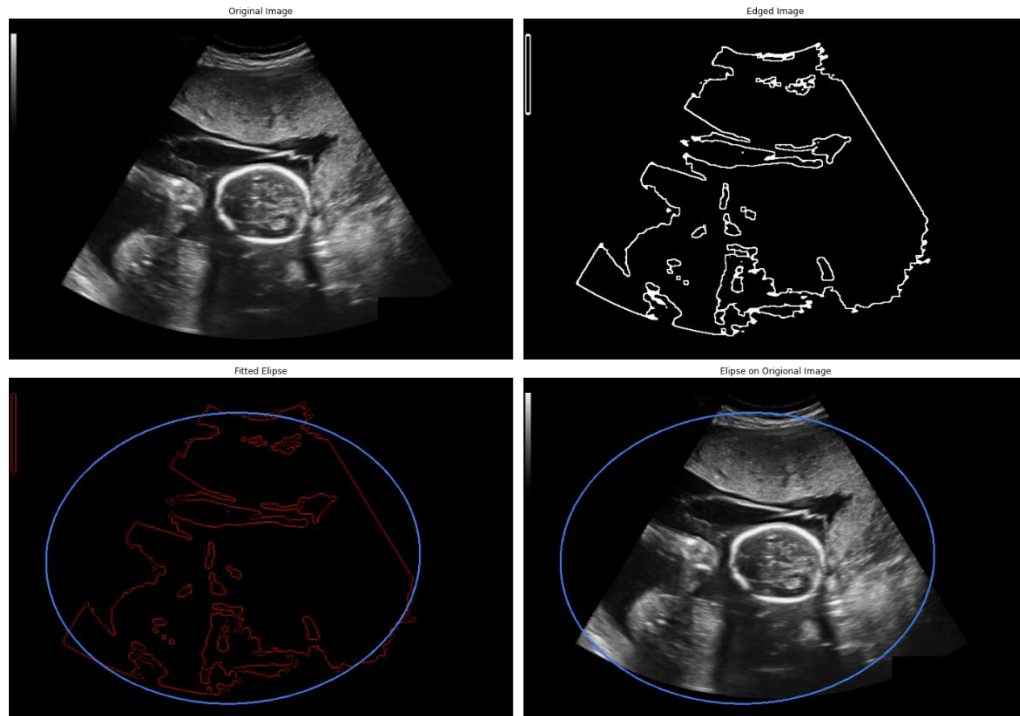


Figure 6: File 637_HC.png – 280% variance

Below is an example of where the algorithm failed to produce a result

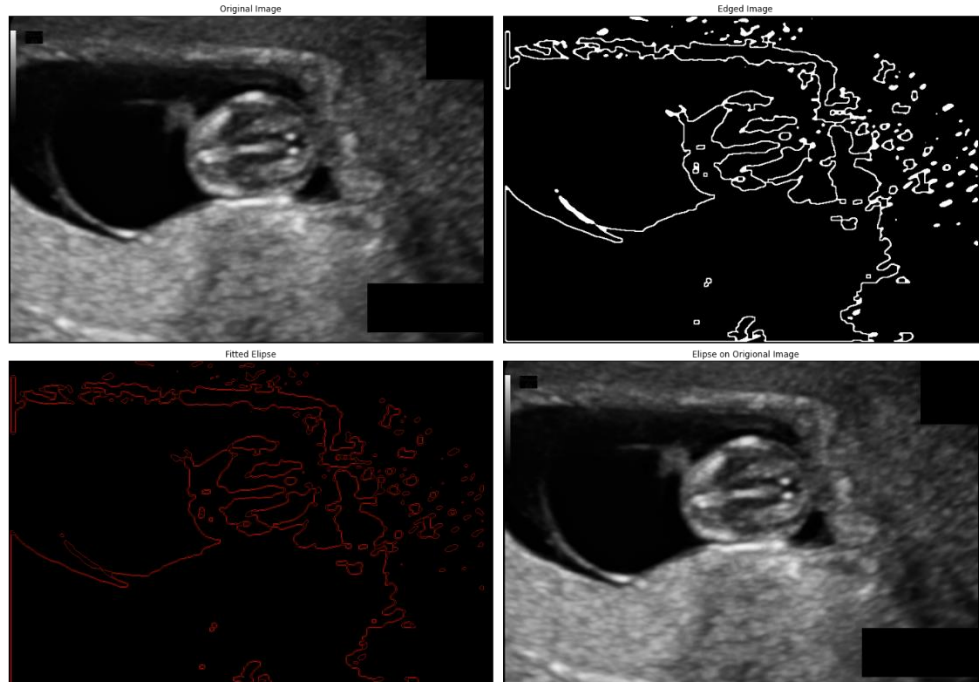


Figure 7: File 001_HC.png

The algorithm produced very good results where the fetal head was clear, image quality was good, and not directly against the uterine wall. As we see in figure 5, the edges are clearly defined and not shared with other structures.

The model completely fails in the instances where the fetal head circumference consists of multiple contours or part of a large contour that represents the uterine wall. Both failure modes can be seen in figure 7. The fetal head is against the uterine wall and thus shares a contour. This image also has additional contours representing other portions of the fetal head.

The majority of false positives seem to stem from where the fetal head failed to be segmented or represented in any contours. This results in the fitted ellipse usually representing either the uterine wall or the edge (frame) of the image if the contour was fairly contiguous. As we can see in figure 6, the fetal head was not represented by a contour and thus it fit the outside ultrasound frame.

Whereas this cannot be considered a success, I am happy with the initial results. My initial expectations were set very low as all the successful models in the competition utilized modern neural networks. As pointed out, 'intensity and gradient-based methods showed a lower performance, due to the fact that they focused more on the appearance of the objects of interest, which indicate high variability' (Meigurger) as opposed to machine learning approaches which dominate the leaderboard in the Grand Challenge. Specifically, I am impressed with how well and accurately the ellipses were placed when a well-defined contour was identified. There are numerous areas that need to be fleshed out, refined, and problems to be resolved.

Future Efforts

There are a few areas that I would pursue to continue improving this model. The first would be to perform a grid search on the filter sizes for erosion and dilation to find the optimal combination. I used a very small number of iterations to find the current combination of filters. A more rigorous search should provide better outcomes for more well-defined edges.

I would also update the ellipse perimeter approximation to one that is more exact, possibly the Ramanujan approximation. The current calculation has an accuracy within 5%. Using a more exact calculation should reduce that variance.

One of the issues noted previously is when the fetal head is part of a larger contour, the model is unable to fit an ellipse to that contour. Below we can see an instance where the model fit an ellipse to a smaller contour and completely missed the larger ellipse (figure 8). This behavior is being driven by how the fitEllipse function is searching. It takes into account the entire contour and not subsections. I experimented (lightly) with subsections of the largest contour. This produced promising results, as seen in figure 9. I would need to refine and optimize the search algorithm. This be used either concurrently with the base model or if an ellipse was not found.

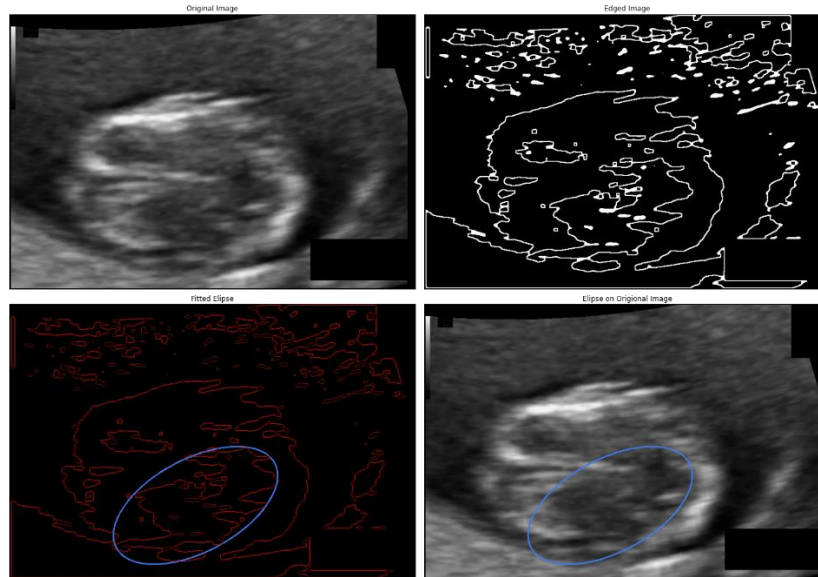


Figure 8: Misplaced Ellipse 33% variance - 022_HC.png

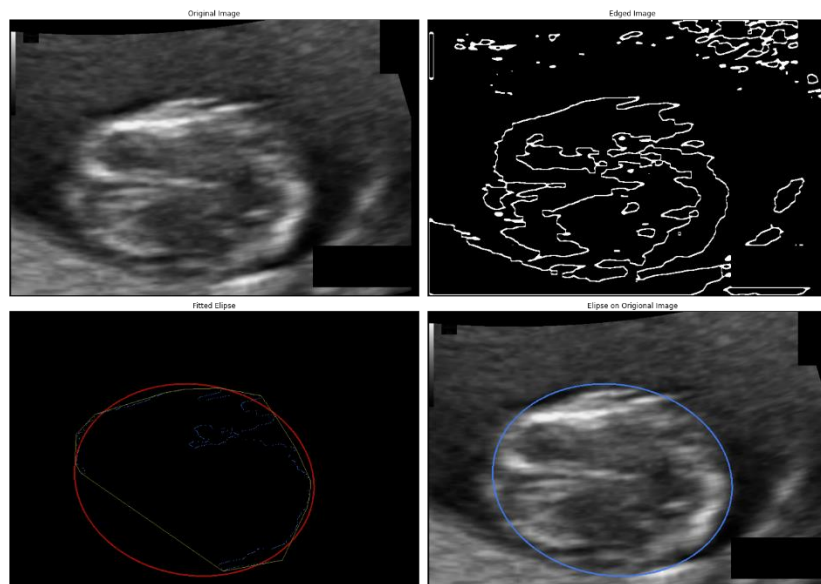


Figure 9: Correctly placed ellipse with sub contours 0.11% variance

I would also like to add more evaluation functionality. Eccentricity would be an excellent attribute to evaluate against. A Fetal head will have an eccentricity closer to zero than to one. More research would be needed to identify what threshold value should be used.

The model needs to be updated to capture all ellipses that are found. Currently it will only return the last ellipse found, although it does draw all ellipses that are found. This would be rolled into the entire evaluation function and ultimately only return, and draw, the most fit.

References

(Thomas 1) Thomas L. A. van den Heuvel, Dagmar de Bruijn, Chris L. de Korte and Bram van Ginneken. Automated measurement of fetal head circumference using 2D ultrasound images. *PloS one*, 13.8 (2018): e0200412.

(Thomas 2) Thomas L. A. van den Heuvel, Dagmar de Bruijn, Chris L. de Korte and Bram van Ginneken. Automated measurement of fetal head circumference using 2D ultrasound images [Data set]. Zenodo. <http://doi.org/10.5281/zenodo.1322001>

(Zeng et al) Zeng Y, Tsui PH, Wu W, Zhou Z, Wu S. Fetal Ultrasound Image Segmentation for Automatic Head Circumference Biometry Using Deeply Supervised Attention-Gated V-Net. *J Digit Imaging*. 2021 Feb;34(1):134-148. doi: 10.1007/s10278-020-00410-5. Epub 2021 Jan 22. PMID: 33483862; PMCID: PMC7887128.

(Gil) Gil Epshtain (<https://math.stackexchange.com/users/529131/gil-epshtain>), What is the parametric equation of a rotated Ellipse (given the angle of rotation), URL (version: 2018-05-06): <https://math.stackexchange.com/q/2647450>

(Pierce) Pierce, Rod. "Perimeter of Ellipse" Math Is Fun. Ed. Rod Pierce. 6 Jan 2022. 11 Nov 2022 <http://www.mathsisfun.com/geometry/ellipse-perimeter.html>

(Meigurger) Meiburger KM, Acharya UR, Molinari F. Automated localization and segmentation techniques for B-mode ultrasound images: A review. *Comput Biol Med*. 2018 Jan 1;92:210-235. doi: 10.1016/j.compbiomed.2017.11.018. Epub 2017 Dec 6. PMID: 29247890.