Checklist for Adult Sponsor (1)

This completed form is required for ALL projects.

To be completed by the Adult Sponsor in collaboration with the student researcher(s):

Student's Name(s): Jeremy Bernstein, Aaron Baruch, Jojo Masri

Project Title: Addressing SIDS: Analyzing the respiratory rate of infants using image processing algorithm

1. ☑ I have reviewed the ISEF Rules and Guidelines.
2. ☑ I have reviewed the student's completed Student Checklist (1A) and Research Plan/Project Summary.
3. ☑ I have worked with the student and we have discussed the possible risks involved in the project.
4. ☐ The project involves one or more of the following and requires prior approval by an SRC, IRB, IACUC or IBC:
   - ☐ Humans
   - ☐ Vertebrate Animals
   - ☐ Potentially Hazardous Biological Agents
   - ☐ Microorganisms ☐ rDNA ☐ Tissues
5. ☑ Items to be completed for **ALL PROJECTS**
   - ☑ Adult Sponsor Checklist (1)
   - ☑ Student Checklist (1A)
   - ☑ Research Plan/Project Summary
   - ☑ Approval Form (1B)
     - ☐ Regulated Research Institutional/Industrial Setting Form (1C) (when applicable; after completed experiment)
     - ☐ Continuation/Research Progression Form (7) (when applicable)

Additional forms required if the project includes the use of one or more of the following (check all that apply):

☐ **Humans,** including student designed inventions/prototypes. (Requires prior approval by an Institutional Review Board (IRB); see full text of the rules.)
- ☐ Human Participants Form (4) or appropriate Institutional IRB documentation
- ☐ Sample of Informed Consent Form (when applicable and/or required by the IRB)
- ☐ Qualified Scientist Form (2) (when applicable and/or required by the IRB)

☐ **Vertebrate Animals** (Requires prior approval, see full text of the rules.)
- ☐ Vertebrate Animal Form (5A) - for projects conducted in a school/home/field research site (SRC prior approval required.)
- ☐ Vertebrate Animal Form (5B) - for projects conducted at a Regulated Research Institution. (Institutional Animal Care and Use Committee (IACUC) approval required prior experimentation.)
- ☐ Qualified Scientist Form (2) (Required for all vertebrate animal projects at a regulated research site or when applicable)

☐ **Potentially Hazardous Biological Agents** (Requires prior approval by SRC, IACUC or IBC, see full text of the rules.)
- ☐ Potentially Hazardous Biological Agents Risk Assessment Form (6A)
- ☐ Human and Vertebrate Animal Tissue Form (6B) - to be completed in addition to Form 6A when project involves the use of fresh or frozen tissue, primary cell cultures, blood, blood products and body fluids.
- ☐ Qualified Scientist Form (2) (when applicable)
- ☐ The following are exempt from prior review but require a Risk Assessment Form 3: projects involving protists, archae and similar microorganisms, for projects using manure for composting, fuel production or other non-culturing experiments, projects using color change coliform water test kits, microbial fuel cells, and projects involving decomposing vertebrate organisms.

☐ **Hazardous Chemicals, Activities and Devices** (No SRC prior approval required, see full text of the rules.)
- ☐ Risk Assessment Form (3)
- ☐ Qualified Scientist Form (2) (required for projects involving DEA-controlled substances or when applicable)

☐ **Other**
- ☐ Risk Assessment Form (3)

Lisa Runco, PhD	[illegible]	09/01/19
Adult Sponsor's Printed Name	Signature	Date of Review (mm/dd/yy)
914-830-5032	lrunco@nshahs.org	
Phone	Email	

Research Plan
Jeremy Bernstein, Aaron Baruch, Joseph Masri

Addressing SIDS: Analyzing the respiratory rate of infants using image processing algorithm

A. Rationale

Sudden Infant Death Syndrome, or SIDS, is similar to, yet a subdivision of sudden unexpected infant death, or SUID. SIDS is a term assigned to the unexpected death, whether explained or unexplained, of an infant. It is the most predominant cause of death in infants ages 2-4 months (Kinney, Hannah C, and Bradley T Thach, 20 Aug. 2009). In the United States alone, according to the CDC, there were 1,400 deaths from SIDS, about 1,300 deaths due to unknown causes, and about 900 deaths from accidental suffocation and strangulation in bed in 2017 (CDC 13 Sept.2019., Figure 1).

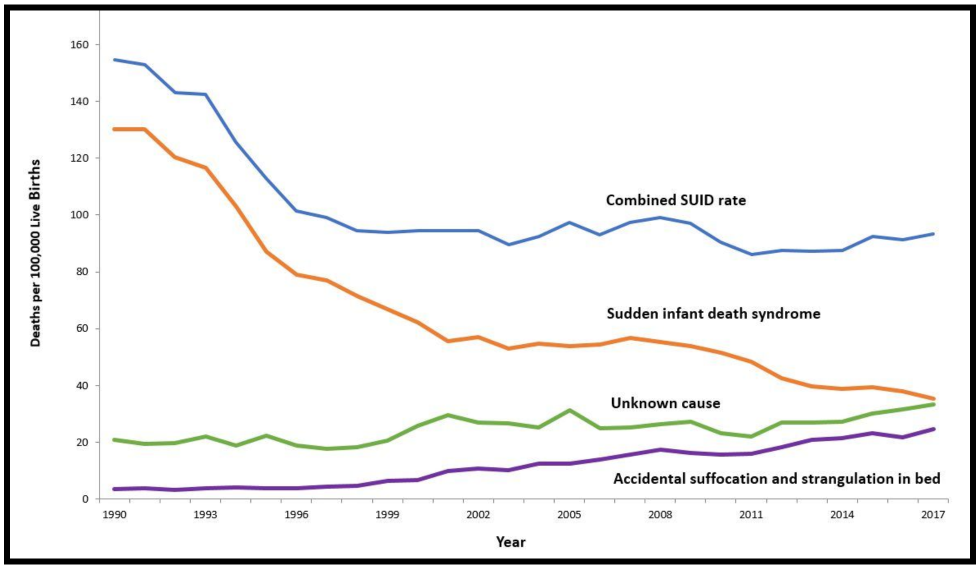
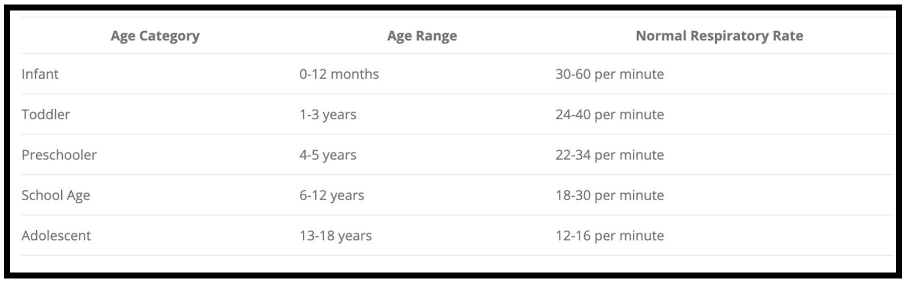


Figure 1: Rate of deaths due to SUID, SIDS, accidental suffocation and unknown causes since 1990 (CDC, 2019).

When unable to declare the infant's cause of death after thorough investigation, the term SIDS is referred to as the cause of death. The sudden death of an infant can also be associated with suffocation, asphyxia, entrapment, infection, ingestions, metabolic diseases, arrhythmia-associated cardiac channelopathies, and trauma, according to the American Academy of Pediatrics (*AAP.org, 2007)*. While the main cause of SIDS is currently unknown, SIDS occurs typically when an infant sleeps on his or her stomach and can occur without warning. It is therefore recommended that children sleep on their backs as opposed to the side or on the stomach to avoid the risk of suffocation during sleep.

The algorithm and technology designed and implemented by The Smart Crib will aim to reduce the amount of deaths due to SIDS. The Smart Crib will continuously monitor baby by incorporating image processing techniques to determine the motion of the chest and if the infant is on its back or stomach using a simple open source facial recognition algorithm. In correlation with a threshold which will be determined with testing and the average breathing rate for infants, which is 30-60 breaths per minute for an infant 0-12 months of age (ACLS Medical Training, figure 2). The app will immediately alert the parents if irregular breathing is detected, therefore saving the infant from suffocation before it is too late. It also has options to use flash or to reverse the camera, as well as an option to disable the camera if not in use for privacy. The app also uses an open source facial recognition algorithm which can detect if the infant is in the crib.



Age Category	Age Range	Normal Respiratory Rate
Infant	0-12 months	30-60 per minute
Toddler	1-3 years	24-40 per minute
Preschooler	4-5 years	22-34 per minute
School Age	6-12 years	18-30 per minute
Adolescent	13-18 years	12-16 per minute

Figure 2: Average Respiratory Rates in Children, Normal Values in Children," ACLS Medical Training

B. Research Questions

1. Is it possible to program a smartphone app to detect breathing in infants?
2. Can we use a smartphone camera to detect if a baby is sleeping on its stomach or not breathing?
3. Based on physiological data regarding infant breathing patterns, can a unique algorithm to detect and decipher normal breathing versus abnormal/not breathing (SIDS) in an infant?

Hypothesis

We can use an image processing machine learning algorithm to determine if an infant is on their stomach or back, and breathing within a normal respiratory rate range.
Using Java, we will build an algorithm to detect breathing in infants and determine a threshold which can be used to determine the breathing rate of an infant and detect irregular breathing.

Engineering goals

To design and program a Processing 3 Java algorithm that, using the Macbook Air 720p FaceTime HD camera, calculates the difference between the amount of pixels in two different images and finding the average movement through change of brightness. The algorithm will then determine if the average pixel count is below or above a certain threshold, the algorithm will identify that an irregular respiratory rate has been detected. After the completion of the algorithm, it will then be imported into a Java application, also made with Processing 3, that will have a user interface for parents to monitor their infants' respiratory rates with their mobile device.

Expected Outcomes

Proof of concept experiment - IBM Watson will be able to detect breathing versus not breathing in this self experiment. A threshold will be determined using the algorithm developed in Processing.

Main outcome - Image processing algorithm will be able to detect the breathing rate in an infant and determine if the infant stops breathing and notify the parents immediately, as well as detect if the infant is in the crib using open source facial recognition.

C. Procedure

Experiment 1

1. For both the control and experimental group data collection, participants will be laying down on a flat surface (cot in the nurse's office) 18 inches away from the laptop webcam The laptop used was an early 2015 Macbook Air with a 720p built-in webcam. There were no windows in the room, and therefore the environment is well controlled in terms of lighting. The data is to be collected at a distance of 18 inches from the participant to the webcam and the webcam will be tilted at an angle of 105 degrees
   At the start of the experiment, student researcher #1 will lie down on the cot. A protractor and tape measure will be used to measure the angle of the webcam and distance from the participant.
2. Student researcher #2 will press run when #1 is ready, and will give directions to #1 as the code collects data through the webcam of the laptop. During the first three seconds (when the code is first run), the participant will be directed to continuously breathe. Data is collected and interpreted by the algorithm during a three second window during a total period of 30 seconds, in 1 second intervals (as described next). 3 clicks of not breathing, 3 clicks of breathing, 15 pictures for each subject (3x15). The entire data collection per participant will take 900 seconds.
3. During seconds 3-6, the participant is told to hold their breath.
4. This process will be repeated so that there will be 15 data collections of “breathing,” and 15 data collections of "not breathing."
5. Experiment is repeated for Student Researcher #2 and #3.
6. The data is then analyzed and a threshold which determines the average breathing rate for a high school student in correlation with the app is determined and eventually scaled down to correlate with the average breathing rate of an infant.

Experiment 2

7. Over the course of two several nights, each student will collect data during their sleep with ambient light in the vicinity. The algorithm will be modified to take pictures and record the average brightness per still every three seconds and will

automatically stop after two hours. The data is used to determine the functionality of the algorithm in low light and the validity of the threshold.

App Development

8. The code, once all tests are successfully completed, will be exported in the form of an Android SDK and exported to an Android mobile device. That device will print the average brightness per frame
 in real time and also allow users to activate/deactivate the camera and flash. The app uses the Ketai processing library which permits the app to use the device's sensors, such as the camera.
9. An SMS algorithm is built using the Processing SMS library and, once tested for accuracy, integrated into the app. The app will ask the user to input a phone number and, if a breathing rate is detected below the threshold, an SMS message will be sent to the user immediately.
10. 45 images of infants lying on their stomach or lying on their back/side that were labeled for reuse were entered into an IBM Watson image recognition data set. 15 of the images were entered into the stomach class, which contained images of infants lying down on their stomachs. A negative class of 15 images was created and images of infants lying down on their back or side were inputted. A test case of 15 additional images of infants lying down on their stomach and back or side was used as a test case to test the IBM Watson data sets.

Algorithm Development

1. Using Processing 3 on the aforementioned Macbook Air, an open source IDE which supports Java and Android app development, an initial algorithm will be developed which uses two stills in a video feed and subtracts the pixel value of both and determining the average brightness of each still which correlates with the amount of movement in each still. The macbook has an Intel Core i5 processor, 4 GB of onboard storage and an Intel integrated HD 6000 graphics processor. Some coding was also done on a homemade PC, which has an AMD Ryzen 2700x processor, MSI MPG X570 wifi-enabled gaming edge ATX motherboard, Nvidia Geforce GTX 1660 ti graphics card, 32GB of Corsair Vengeance Pro 3000mhz DDR4 RAM, 1 TB Samsung Evo SSD and 2 500 MB Crucial M.2 SSDs.
2. The code will take dimensions of the video and saves them into float variables p0, p1 and p2, as well as the RGB values.
3. The RGB values are then constrained to be between 0 and 255.
4. The algorithm will then use these dimensions to calculate the pixel density within the stills of the video.
5. The determined value is then printed in a screenshot which is taken every three seconds automatically. An Alternative version of the code was made which will only produce a screenshot when the mouse is clicked.
6. The algorithm is then exported as an Android SDK using the Kentai library to access the camera. Additional code is added for flash as well.
7. An SMS function is added using the Processing SMS library.

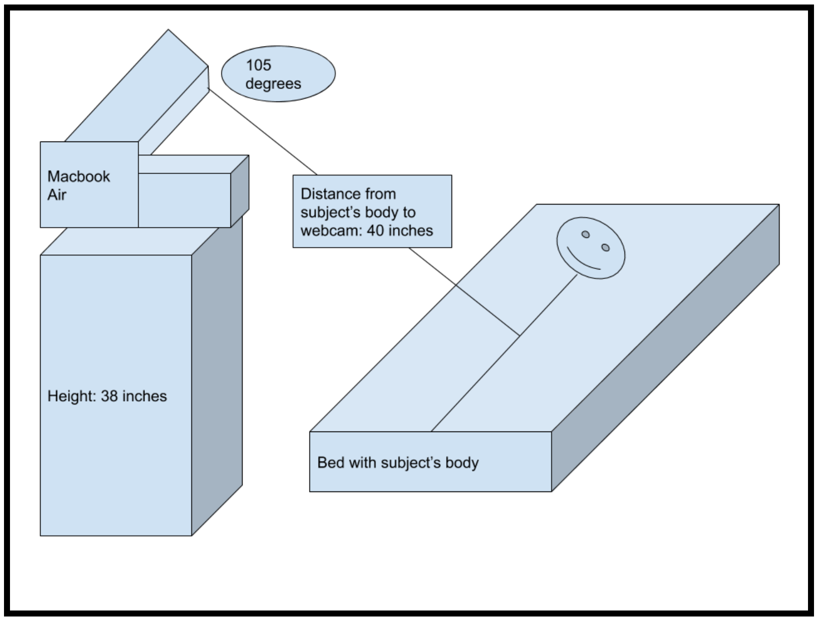


Figure 3: Diagram to illustrate how data was collected during the first trial, image created by Jeremy Bernstein

Code Developed

```
		int clickValue = 0;
int nameVal = 0;
int nameValAlso = 0;
float t0;
float t1 = millis();
// this version is to create threshold for not just positive change in pixel intensity, but also magnitude threshold as well.
// this version features pixel error instead of summing every pixel's error together.
// the thought behind the previous version is that if we can detect any change at all, then we can zoom in later. But why spend the extra to zoom in if there is no change in the first place.
// if this doesn't work in complete darkness, which is might even just from some led light from laptop /phone, but if needed, could use ir light add on or something part of unversal clip.
int threshold = 100; // 10 looks good, fun to play with other values though.
// Step 1. Import the video library
import processing.video.*;
// Step 2. Declare a Capture object
Capture video;
float[] p1;
float[] p0;
float[] p2;
 float p3=0;
int loopcount;
float a;
void setup() {
  size(320, 240);
  p1 = new float[width*height];
  p0 = new float[width*height];
  p2 = new float[width*height];
  // Step 3. Initialize Capture object via Constructor
  video = new Capture(this, 320, 240);
  video.start();
}
// An event for when a new frame is available
void captureEvent(Capture video) {
  // Step 4. Read the image from the camera.
  video.read();
}
void draw() {
  loadPixels();
  video.loadPixels();

  for (int x = 0; x < video.width; x++) {
    for (int y = 0; y < video.height; y++) {
      // Calculate the 1D location from a 2D grid
      int loc = x + y * video.width;

      // Get the red, green, bl-ue values from a pixel
      float r = red  (video.pixels[loc]);
      float g = green  (video.pixels[loc]);
      float b = blue  (video.pixels[loc]);


      // Constrain RGB to make sure they are within 0-255 color range
      r = constrain(r, 0, 255);
      g = constrain(g, 0, 255);
      b = constrain(b, 0, 255);
      p1[loc]=r+g+b;
      if (p1[loc]-p0[loc]>threshold) {
        p2[loc] = p2[loc] + p1[loc]-p0[loc];
      }
      noStroke();
      fill(p1[loc]-p0[loc]);
      rect(x, y, 1, 1); // since there are two feeds we want to have an off set that starts half way down the screen
      p0[loc]=p1[loc];
    }
  }

  for (int i = 0; i< video.height*video.width; i++) {
    p3=p3+p2[i];
  }
  print("total positive brightness change is "); println(p3/(video.height*video.width)); p3=0;
  for (int i = 0; i< video.height*video.width; i++) {
    p2[i]=0;
  }
  // next we need to print them all and reset all the p2's. using for loop loc
  nameVal = ++nameValAlso;
    //nameVal.Integer.toString(int);
    // uncomment below to save images

  }
void keyPressed() {
  if (clickValue == 0) {
    clickValue = 255;
    saveFrame("breathing" +nameVal+".jpg"); //change string in quotes to "not breathing"
    nameVal++;
  }
  else
  {
    clickValue = 0;
  }
}
/*
// Step 1. Import the video library
import processing.video.*;
float ppp;

// Step 2. Declare a Capture object
Capture video;
float rt, gt, bt = 0;
float[] p1;
float[] p0;
float[] p2;
int loopcount;
float a;
void setup() {
  size(320, 240);
  p1 = new float[width*height];
  p0 = new float[width*height];
  p2 = new float[width*height];// p0 previous p1 current p2 is edited/computervision
  // Step 3. Initialize Capture object via Constructor
  video = new Capture(this, 320, 240);
  video.start();
}

// An event for when a new frame is available
void captureEvent(Capture video) {
  // Step 4. Read the image from the camera.
  video.read();
}
void draw() {

  loadPixels();
  video.loadPixels();



  for (int x = 0; x < video.width; x++) {
    for (int y = 0; y < video.height; y++) {
      // Calculate the 1D location from a 2D grid
      int loc = x + y * video.width;

      // Get the red, green, blue values from a pixel
      float r = red  (video.pixels[loc]);
      float g = green(video.pixels[loc]);
      float b = blue (video.pixels[loc]);


      // Constrain RGB to make sure they are within 0-255 color range
      r = constrain(r, 0, 255);
      g = constrain(g, 0, 255);
      b = constrain(b, 0, 255);
      p1[loc]=(r+g+b)/3;
      noStroke();
      fill(p1[loc]-p0[loc]);
      ppp= ppp+p1[loc]-p0[loc];

      rect(x, y, 1, 1); // since there are two feeds we want to have an off set that starts half way down the screen
      if (loc>=76760) {
        ppp=ppp/video.width/video.height;
        print("ppp is ");
        println(ppp);
        ppp=0;
      }
      p0[loc]=p1[loc];
      // print("loc is ");
      // println(loc);
    }
  }
  t1 = millis()-t0;
  // println(t1);
  if (t1>3000) { // this is the threshold, 10000 means every 10 sconds take photo
    t0=millis();
    nameVal = ++nameValAlso;
    //nameVal.Integer.toString(int);
    // uncomment below to save images
    //  saveFrame("breathing" +nameVal+".jpg"); //change string in quotes to "not breathing" for not breathing data set
  }
}
*/
```

Computer angle = 70 degrees
Table height = 23 inches
Height of laptop at 70 degrees = 37.5 inches
Distance from computer angle to object = 36 inches
Height of the stool = 14.5

Risk Analysis
None

Supervision:
All experimentation was approved and monitored by Dr. Lisa Runco (Director of Science Research, North Shore Hebrew Academy High School (NSHAHS)), Mr. David Weinberg (Chair of Engineering Department, NSHAHS) and/or Mr. Peter Suchmann (Coordinator of Science Research, NSHAHS)

Data Analysis
Used IBM Watson to determine if an image recognition algorithm could detect breathing in an image captured using the Java Processing algorithm. The images were added into two classes, the breathing and negative (not breathing) classes, and then were tested using sample images. Each result returned correctly with a confidence of 0.9 or above which confirmed that this project was viable.

D. Bibliography

1. Kinney, Hannah C, and Bradley T Thach. “The Sudden Infant Death Syndrome.” The New England Journal of Medicine, U.S. National Library of Medicine, 20 Aug. 2009, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3268262/
2. "What causes SIDS?". National Institute of Child Health and Human Development. 12 April 2013. Archived From the original on 2 April 2015. Retrieved 9 March 2015.
3. “Normal Values in Children.” ACLS Medical Training, ACLS Medical Training, www.aclsmedicaltraining.com/normal-values-in-children/.
4. “Data and Statistics for SIDS and SUID.” *Centers for Disease Control and Prevention*, Centers for Disease Control and Prevention, 13 Sept. 2019, www.cdc.gov/sids/data.htm.
5. “Sudden Infant Death Syndrome (SIDS).” *Mayo Clinic*, Mayo Foundation for Medical Education and Research, 13 Nov. 2018, www.mayoclinic.org/diseases-conditions/sudden-infant-death-syndrome/symptoms-causes/syc-20352800.
6. “Safe Sleep.” *AAP.org*, www.aap.org/en-us/advocacy-and-policy/aap-health-initiatives/healthy-child-care/Pages/Safe-Sleep.aspx.

NO ADDENDUMS EXIST