Using Perceptual Grouping and Segmentation to Improve Aviation Maintenance

CS50100 Computing for Science and Engineering

Fall 2018

Final Report

November 30, 2018

Team Members:

Javier Belmonte (Mechanical Engineering, belmontj@purdue.edu),

Xin Ding (Civil Engineering, ding245@purdue.edu),

Maria Kon[*] (Psychological Sciences, mkon@purdue.edu),

Laith Sakka (Electrical and Computer Engineering, lsakka@purdue.edu),

William Weldon (Technology, wweldon@purdue.edu).

[*] Team leader.

Table of Contents

1. Introduction

Title 14 of the Code of Federal Regulations part 43 section 13 (14 CFR 43.13) describes the procedures by which maintenance on aircraft is performed. This regulation states that maintenance will be done in accordance with the manufacturer's maintenance requirements. One element of all aircraft manufacturer specifications is replacement of damaged rivets, which are described in Advisory Circular 43.13-1B "Acceptable Methods, Techniques, and Practices." Most aircraft contain hundreds to hundreds of thousands of rivets which makes the human inspection of every rivet on an aircraft a functional impossibility. In order to provide this capability we will use grouping theories from psychology with a focus on the detection of cracked and smoking rivets.

Using a robotic data collection platform in concert with perceptual grouping techniques has the potential to allow for autonomous rivet detection. The first, and biggest, challenge to implementing this system is the subtle variation in rivet distances in both the x and y axes in relation to each other. This variation prevents the system from comparing the images it collects with a template in order to determine placement and damage. The second challenge is the variable nature of aircraft wings. Each aircraft type has a different airfoil and wing shape which would require each aircraft to have a custom path created, and while this is possible it increases the requirements on operations personnel.

We have labeled the combination of these two problems "The Rivet Analyzer Problem". In order to solve this problem we will create an autonomous system capable of autonomously mapping rivet positions on the aircraft as well as detecting damage.

We approach the Rivet Analyzer Problem by considering how humans are able to scan rivet rows and identify rivet damage. Maintenance technicians visually inspect the surface of the aircraft, and group elements of this scene. These groups are then processed into objects, individual rivets, and patterns, rows of rivets. In psychology, this task is said to involve perceptual grouping. Generally, perceptual grouping is how humans determine which elements in a visual scene form groups such as objects and patterns.

In psychology there are rules, e.g., the Gestalt grouping rules (Wagemans et al., 2012), and theories (for a recent overview, see Dresp-Langley, Grossberg, & Reeves, 2017), that attempt to give some account of grouping. However, most of these accounts are at an abstract level and, thus, do not offer a concrete, mechanistic account of grouping. However, there is one recent neural network of the visual system that provides a mechanistic account of one type of grouping effect, namely crowding (see Section 1.2.2 for details). This neural network is the version of the LAMINART model implemented by Francis et al. (2017). Unlike other computational models used to simulate such behavior, LAMINART includes a grouping component (see Doerig et al., 2018, for a direct, detailed comparison of the performance of LAMINART with that of thirteen other computational models). Moreover, LAMINART may offer a means of providing a mechanistic account, rather than merely an abstract theory, of grouping in terms of segmentation.

However, the 2017 version of LAMINART is currently very limited. It is tailored to stimuli that feature a target in the center and two groups of non-target elements on either side, and it only offers a mechanistic account of the particular grouping effect of crowding. A row of rivets may not feature a central target, e.g., the damaged rivet may be near the edge of a visual scene. Indeed, a human inspector must visually search for damaged rivets; the inspector cannot simply assume that the damaged rivets will be in the center of the visual scene and flanked on either side by undamaged rivets.

Thus, we are solving two interrelated problems: The Rivet Analyzer Problem, which is stated above, and The Grouping Problem. The Grouping Problem is the problem of creating and implementing a biologically-based circuit in an existing neural network of the visual system such that the network can group elements of a visual scene, e.g., rivets in a photo of an aircraft wing, into objects, e.g., undamaged rivets, and patterns, e.g., rows of rivets.

1.1 Why these problems are important

Both of these problems are important to resolve for the following reasons.

1.1.1 The importance of resolving the Rivet Analyzer Problem

Rivets are the main fasteners used on aluminum alloy aircraft, and during an airframe inspection these rivets must be assessed. Any passenger jet contains tens of thousands of identical rivets. Creating a vehicle to follow a rivet row and gather visual data on the rivets, and track the rivet positions, would allow aircraft manufacturers and operators to accurately map their positions. Autonomously mapping rivets would allow subsequent maintenance visits to compare rivet conditions and show how the fasteners wear over time. This information has the potential to assist in the creation of airframe digital twins by creating a map of the rivet positions as they exist in the aircraft, instead of the positions they should be in.

1.1.2 The importance of resolving the Grouping Problem

Much experimental work has been done in psychology on perceptual grouping, i.e., how humans determine which elements in a visual scene form groups such as objects and patterns. However, there is not a well-developed mechanistic account of grouping. For example, although LAMINART has a viable collection of circuits for outputting information about boundaries and luminance, it currently lacks a well-developed, biologically-based way of segmenting and grouping this information for phenomena beyond crowding. Note that by ‘a biologically-based way’, we mean a way that is at least consistent with the results of psychophysical experiments.

1.2 Related work

1.2.1 Airframe Digital Twins

Tuegel (2012) presents some of the challenges to realizing an airframe digital twin (ADT). The first problem that is stated is “Establishing the initial conditions for the ADT,” and this project stands to assist in establishing one part of the initial aircraft condition. This initial condition is the actual condition of the aircraft as it currently exists. This condition is difficult to determine when the aircraft leaves the factory and is nearly impossible to determine after the aircraft has been in service.

One potential partial solution to this problem is to create a 3D model of the aircraft using orthomosaicing methods. These models are as, or more accurate, than models created with LiDAR sensors (Zarco-Tejada, Diaz-Varela, V.Angileri, & Loudjani, 2014). These models have been used for remote inspections on dams (Henriques & Roque, 2015), breakwaters (Henriques, Fonseca, Roque, Lima, & Marnoto, 2014), and after controlled burns (Simpson, et al., 2016). In order to create highly accurate models with orthomosaicing, special equipment and knowledge is required. As aircraft scale increases, inspecting individual rivets will likely prove challenging. The creation of a system to analyze the rivets in an aircraft will allow for accurate mapping of this aircraft component. The result could also be fused with an orthomosaic, or checked against an orthomosaic, in order to improve accuracy of the 3D model created.

1.2.2 Grouping

Grossberg and colleagues, e.g., Grossberg and Mingolla (1985), provide a theory of real-time visual processing the involves two main interconnected circuits: the Boundary Contour System (BCS) and the Feature Contour System (FCS). Given an image, BCS, which creates real and illusory boundaries, and FCS, which 'fills in' luminance within bounded elements, work together to output an image with boundaries and luminance.

One recent model that uses this theory is LAMINART. LAMINART is a neural network. It models bottom-up processes, e.g., activity of cells in the human retina, and incorporates top-down processes, i.e., it uses a segmentation process to model perceptual processes driven by cognition. The version of LAMINART implemented by Francis, Manassi, and Herzog (2017) did introduce a means of grouping elements of an image. Further, it successfully simulated a wide variety of perceptual crowding experiments. Crowding experiments feature stimuli that are presented very briefly. A typical stimulus in a crowding experiment consists of a target in the center with two sets of distractors on either side of the target (e.g., Fig. 1, left panel). The task is to report whether the target is shifted slightly to the left or right. Depending on a variety of factors, e.g., size, shape, and relative location of the distractors, the task can be difficult for humans to perform because the distractors 'crowd' the target. Yet, Francis et al. were able to give a mechanistic account of these crowding experiments using LAMINART's segmented output (e.g., Fig 1, right panel). For the case depicted below, Francis et al. proposed that human participants performed poorly because the target and distractors were not segmented into different groups. Instead, illusory contours (e.g., the green horizontal lines in Fig. 1) form between similarly sized target and distractors, making one perceive them as a single group.

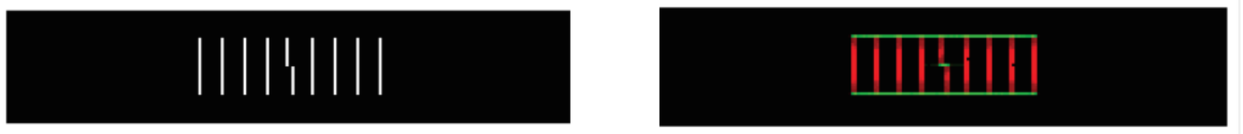


Figure 1. Left: example stimulus from a crowding experiment. See text for details. Right: output from the 2017 version of LAMINART. Each pixel is represented by a neuron in the neural network. The higher the intensity of the pixel, the higher the activation of the neuron. Color represents the preferred orientation of the neuron. (Adapted from Francis et al., 2017, p. 484.)

The Francis et al. (2017) study was limited in two ways.

First, it only simulated a particular set of crowding experiments. In the present project, we ran simulations using the Manassi stimuli as well as an experiment by Palmer and Beck (2007), which has features similar to our Rivet Analyzer task, e.g., experiments in which the stimuli have no central target. While Francis et al. simulated the results of experiments in which participant performance was evaluated, our aim was to produce qualitatively similar results to subjective rating experiments reported in the Manassi and Palmer and Beck papers. In these experiments, participants were asked to rate how well a target stood out in an image. Our rationale for choosing to simulate these experiments is due to the fact that we want the system to optimize groupings of elements such that 'targets', i.e., damaged rivets, stand out from the undamaged rivets.

Second, the circuit that models top-down processes in the 2017 version of LAMINART is relatively undeveloped. In particular, its selection signals are, effectively, two semi-randomly located circles of a single diameter placed on left and right sides of the image. So, although this version of LAMINART has successfully simulated the results of some crowding experiments, its novel segmentation process is very limited. In turn, development of LAMINART's representation of top-down processes requires development in order for the model to be applicable to a wider range of visual phenomena, e.g., images of bespoke rows of rivets on an aircraft wing.

1.3 Contributions to aviation maintenance and psychology

This project has two main contributions. First, we developed a system consisting of a vehicle that can: follow a row of rivets on an aircraft wing, not fall off the wing, detect damage over the row of rivets, and map the rivet positions in relation to the other rivets. As discussed below, this system works for particular types of rivet damage and, depending on the type of damage, we can flag damage by rivet or by frame. Second, we have worked on developing a biologically-informed circuit that points towards a mechanistic account of grouping.

More specifically, have gone towards developing means of turning a Lego EV3 robot into a mobile data collection platform capable of safely traversing an aircraft. In order to analyze the raw data collected via the robot, we developed:

- a program that gets the robot to follow a line of rivets by using a luminance detector that follows a black line that is parallel to the rivets;
- a classifier that is trained to detect damage and undamaged rivets from a training set of rivets that we collected;
- a pre-processing program, which converts raw video footage into images that are of a suitable size, rotation, color and format, as input for LAMINART;

- LAMINART's segmentation process in a biologically-informed manner such that it maps the relative position of rivets, indicates some damage per frame, and simulates results of a behavioral study;
- a GUI that integrates the pre-processing program and our version of LAMINART;
- two error detection programs:
    - one gives a calculation of the difference between a target and actual image and indicates whether there is at least one type of damaged rivet in the frame, and
    - the other can detect whether an individual rivet is cracked plus create 'target' images, i.e., images that have undamaged rivets in the same locations as those in the actual image.
- some components of post-processing, namely, code that maps output from each of LAMINART'S segmentation layers back onto the preprocessed image, and code that rotates the pre-processed images from a diagonal to a vertical orientation.

Note that we did not complete our task of writing a program that pieces together the output of LAMINART into a 'virtual scan' of a row of rivets that indicates each rivet's relative position, which rivets are damaged, and the type of damage each damaged rivet has. This was due to two main factors. First, we encountered several issues in data collection, e.g., lighting conditions that could not be corrected in pre-processing, difficulty finding rows with actually damaged rivets that were the same type of flat headed rivet as those on the particular aircraft from which we collected our raw data. Such issues required us to spend more time than anticipated in the data collection and pre-processing stages of our project. Second, we discovered that when attempting to stitch together our overlapping images, our program either skipped or duplicated rivets. This is due to the general homogeneous appearance of the rivet rows. There does not seem to be enough information in each image for this strategy to work. Nevertheless, pending the development of an alternative algorithm to piece together the output from LAMINART and the integration of our error detection techniques, the task of writing a program that stitches together the output of LAMINART into a 'virtual scan' of a row of rivets that indicates each rivet's relative position, which rivets are damaged, and the type of damage each damaged rivet has, seems to be an achievable goal. Thus, with future work on the post-processing stage, we anticipate that this system will be able to 'scan in' a row of rivets on an actual aircraft and provide information about rivet damage. Thus, we have at least gone towards a system that can contribute towards the broader goal of creating an aircraft digital twin.

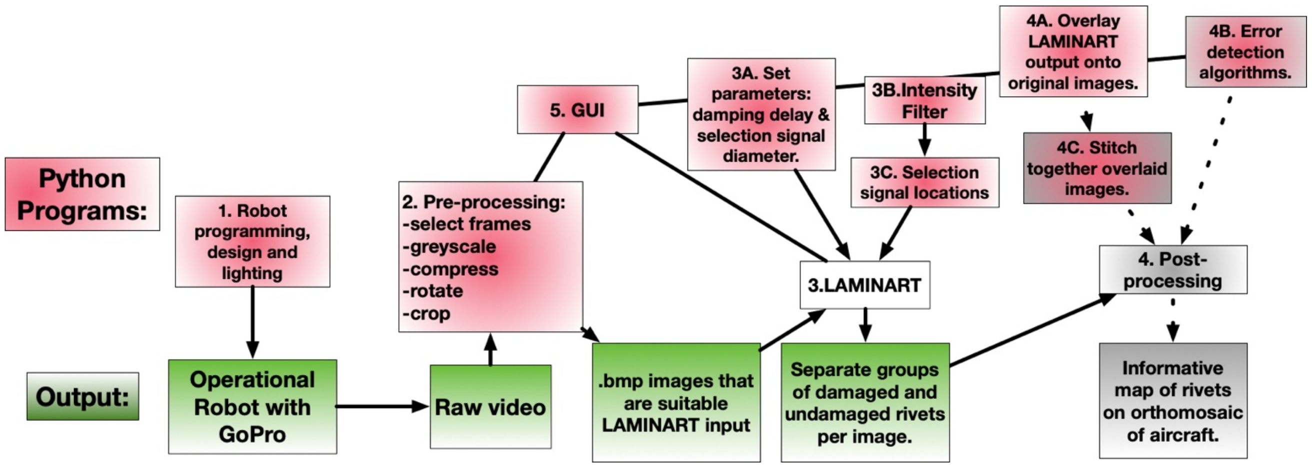
1.4 Organization

This report is organized as follows. In Section 2 we describe the materials and methods we used to solve the problem outlined in Section 1. Specifically, we describe: the modified Lego robot, pre-processing programs, the LAMINART neural network, post-processing components, and the GUI we developed. Section 3 provides the results of our system and analysis of these results. This section is broken into subsections corresponding to our project's four main subsystems, namely, data collection, pre-processing, LAMINART, post-processing, plus the GUI. Section 4 discusses the progress, difficulties and limitations, of the subsystems laid out in Section 3. We also discuss potential future improvements for this study in this section. Section 5 describes the conclusions we draw from this test. Section 6 provides acknowledgments to those who allowed us to use their equipment or provided us with other assistance during the process. Section 7 provides our references in APA format.

2. Materials and Method

To solve the problems detailed in Section 1, we fed compressed images from a GoPro camera attached to a Lego EV3 robot into a computational model of human visual perception called LAMINART. The output of LAMINART was used to point towards solutions to the rivet analyzer and grouping problems specified above.

More specifically, the main components of our subsystem are schematized in Fig. 2:



Python Programs:
Output:
1. Robot programming, design and lighting
Operational Robot with GoPro
Raw video
2. Pre-processing: -select frames -greyscale -compress -rotate -crop
5. GUI
.bmp images that are suitable LAMINART input
3A. Set parameters: damping delay & selection signal diameter.
3B.Intensity Filter
3C. Selection signal locations
3.LAMINART
Separate groups of damaged and undamaged rivets per image.
4A. Overlay LAMINART output onto original images.
4B. Error detection algorithms.
4C. Stitch together overlaid images.
4. Post-processing
Informative map of rivets on orthomosaic of aircraft.

Figure 2. Schematization of our approach to the Rivet Analyzer and Grouping Problems.

The main tasks that require coding are numbered 1 through 5. The red boxes generally indicate programs that we have completed and written in Python to do the operations specified in them. The green boxes indicate the main output from the program. Grey boxes indicate subsystems and output the we proposed to produce, yet were unable to complete in time for this report. Boxes that are mixes of red and grey indicate that, although we have made some progress on the module, it is not yet in working condition to be incorporated into the post-processing stage. Further, note that the main LAMINART code, indicated by the white box, was modified such as to incorporate the module in 3B, but it was largely written by one of the group member's supervisor (G. Francis).

The following subsections provide more detail regarding each subsystem. The subsection numbering follows the subsystem numbering in Fig.2.

2.1 Robot and raw data collection

The data collection platform chosen for this project was a Lego EV3 robot. This vehicle was chosen out of convenience, but also provided the necessary functionality for this test. The vehicle was provided with a new OS, ev3python, in order to run on Python.

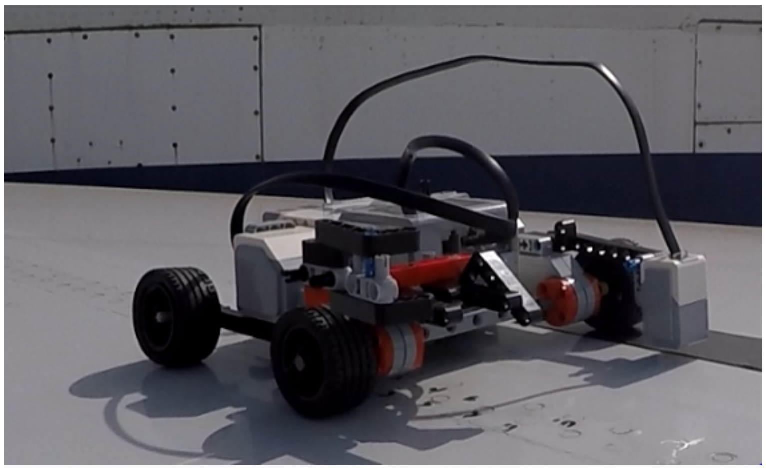


Figure 3. Robot Iteration 1.

Alongside the software challenges, we faced hardware challenges with the data collection platforms. These challenges led to multiple iterations of the platform. This first iteration worked well for initial test data but held the camera too close to the target, preventing the camera from focusing. This led to iteration two, which had the camera set higher up on the right side of the vehicle. The camera was not offset far enough to allow for focusing and was updated to iteration three within an hour.

Iteration three, shown in Fig.4 below, saw the camera mount shifted to the center of the vehicle, and a capability to have a light mounted to the camera rig. This iteration requires a counterweight when equipped with the camera and light, and a counterweight location has been integrated. This version allowed the camera to correctly focus on the target, rivets, but the light appears to be ineffective for processing.

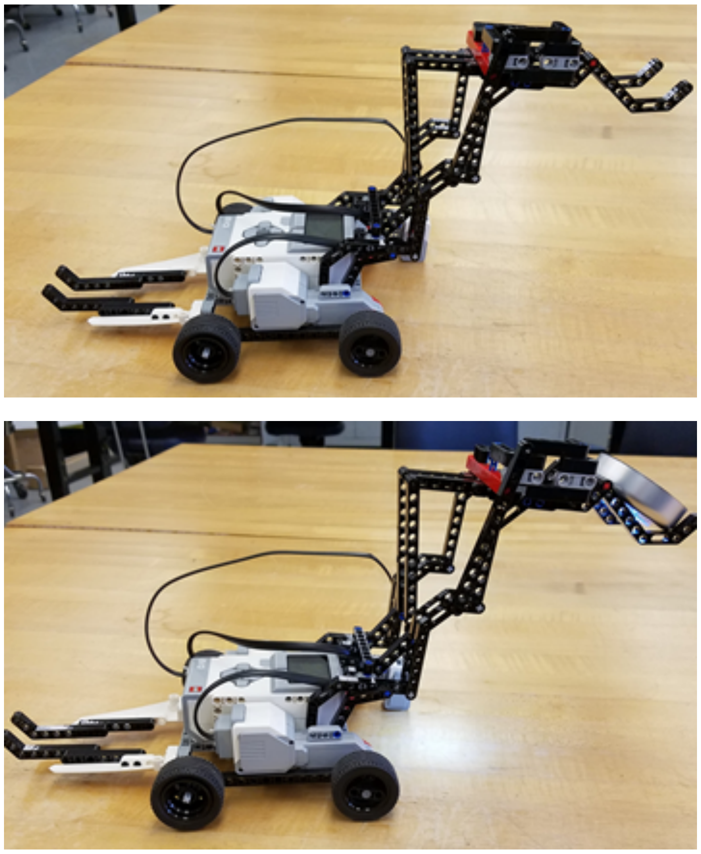


Figure 4. Robot iteration 3.

2.2 Image pre-processing

We defined image collection techniques. Additionally, we used output from the pre-processing system to determine how to appropriately setup the GoPro camera on the robot so that it takes usable video.

In the process of defining image pre-processing techniques, we determined how to split up and compress video from the GoPro camera such that the resulting images are suitable for our purposes. By ‘suitable’, we mean that the image consists of only relevant information (i.e. only the rivets will be shown, and a total of three rivets pairs worked best), and the image must be small enough that LAMINART can process it within a reasonable amount of time yet is detailed enough to be informative regarding rivet damage. Additionally, to make use of the diagonal 45-degree grouping direction of LAMINART the final output of pre-processing was designed to show rows of rivet with that orientation. The number of frames extracted from the video was decided such that there are enough frames to analyze every rivet without repeating them.

In order to achieve the desired input for LAMINART, the pre-processing algorithm was divided into different steps that preformed a task to achieve the overall result. An Object-Oriented Programming approach was used to develop the pre-processing algorithm, taken advantage mainly of the design, encapsulation, and software maintenance benefits of this programming paradigm. The stages of the algorithm, alongside with their descriptions, can be seen in the following lines:

I. Frames extraction: Frames are extracted at desired intervals, called a frame step, based on the length of the image captured by the robot and its speed. In such a manner, it was achieved to extract only frames that do not have repeating rivets.
II. Grayscale conversion: Many of the following methods used in pre-processing require the use of images consisting only in gray tones. A standard method of OpenCV was implemented to achieve this.
III. Rivets area cropping: Images extracted by the robot consist of more than only rivets, since our interest lies in the rivets, an algorithm was developed to identify where they are located within a frame and then crop the picture based on it. To accomplish this requirement, we took advantage of the circular shape of rivets by creating a program that uses Hough transform to detect circles and crop the image based on the obtained positions. The circles are drawn in a separate image that is considered "ideal" and that servers as a target for comparison in the error detection code used in post-processing.
IV. Brightness and contrast enhancing: To allow for a more controlled definition of the frames a function to alter and improve contrast and brightness was designed. Two different approaches were developed and tested. These methods worked on histogram equalization, and multiplication and addition, respectively.
V. Black and white conversion: Ultimately, LAMINART's input needs to be in the form of black and white, therefore different approaches to turn a picture into black and white were implemented. The algorithms used are adaptive thresholding and Otsu's thresholding. This frame also serves as input for the error detection code used for post-processing.
VI. Resizing: LAMINART needs input that is small enough to allow for reasonable computing time, thus a resizing stage was included in pre-processing. Area interpolation was the decided method to resize images, with care on preserving the aspect ratio obtained in the cropping step.
VII. Rotation: As mentioned, LAMINART's grouping signals are 0, 45, and 90 degrees. To take advantage of the 45-degree grouping signal the frames are rotated so that the row of rivets aligns with this direction. An algorithm that used linear regression to obtain a line fit for the position of the center of the rivets was devised, and the slope of this line converted to the degrees by which the frame needs to be rotated to obtain the desired orientation. Furthermore, special attention was given to the rotated image to prevent loss of rivets by parts of the image that are left out of bounds in rotations.

Given the many stages of pre-processing and the variations between the images contained in different videos, it was thought that a method to quickly test and show results of these stages would be useful. Consequently, a such a method was included in the pre-processing class. This allowed for inspection of the outcome of every stage in the algorithm and the consequent feedback to data collection and pre-processing techniques.

Additionally, an orthomosaic was created from the initial images. We planned for the orthomosaic to allow us to examine the entire rivet row from one image and allow us to overlay the LAMINART output for comparative purposes. However, since we did not finish the post-processing system, we were not able to overlay the output on the orthomoasic.

2.3 Develop LAMINART

We developed and coded a biologically-based grouping circuit using Python that allows LAMINART to perform the grouping tasks necessary to analyze rivets. To ensure that it is biologically-based, rather than merely an *ad hoc* manipulation of LAMINART for this particular task, the grouping circuit was developed by simulating human performance in psychophysical experiments that feature grouping tasks.

2.3.1 The 2017 version of LAMINART

LAMINART is a neural network that provides a model of how visual information is processed in various layers of the visual cortex. It consists of neurons representing neuronal activity in the primary visual cortex (V1) and the secondary visual cortex (V2). LAMINART's neurons, similar to those of the visual cortex, belong to one of three distinct layers in V1 and V2. Each layer is labelled numerically by: 6, 4 and 2/3 (see Fig. 5). This part of the neural network operates by having a neuron represent each pixel in the stimulus. The activity of this neuron is a function of the orientations and luminance of pooling neurons and controlled by AND-gates created by the black interneurons (see Francis et al., 2017, p. 486, for further details). This part of the model outputs images featuring real and illusory contours in which each pixel represents the output of a neuron (namely, that represented by the middle white circle in the 2/3 layer of V2 in Fig.5), and the intensity of the color indicates the numbers of spikes generated by the model neuron over 50ms (see Fig. 1 above). Thus, the higher intensity of a pixel, the more active the neuron is.

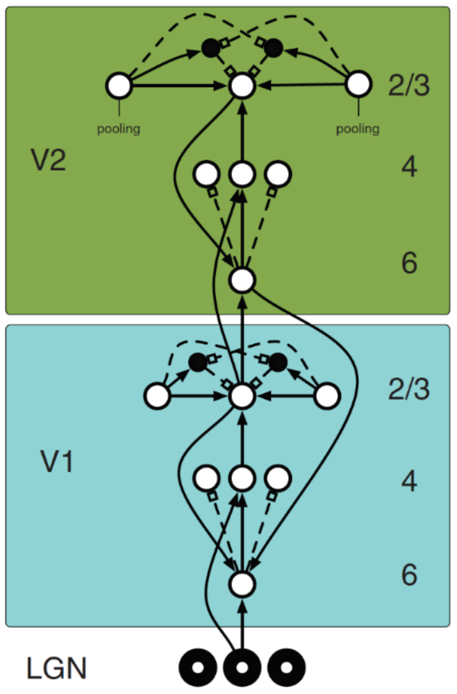


Figure 5. Depiction of the layers used in the 2017 version of LAMINART. The central column of open circles, which are model neurons, represents neurons at a single retinotopic position. Black circles represent interneurons that form AND-gates with model neurons. Arrows represent excitatory connections, while dashed lines with square heads represent inhibitory connections. Donuts at the bottom represent inputs from the lateral geniculate nucleus. (Image from Francis et al., 2017, p. 486.)

In addition to representing neuronal activity, this version of LAMINART also has a top-down mechanism, i.e., it uses a segmentation process to model perceptual processes driven by cognition. The segmentation process involves two selection signals. Each signal is a circle of a specified diameter, with one placed to the right of the center of the stimulus and the other to the left. If the signal is placed over pixels with some intensity, i.e., over activated neurons, this signal will spread to neighboring active neurons. Such connected neurons may move to different segmentation layers (see Fig.6). The content of each segmentation layer is considered to be a possible grouping of stimulus elements.

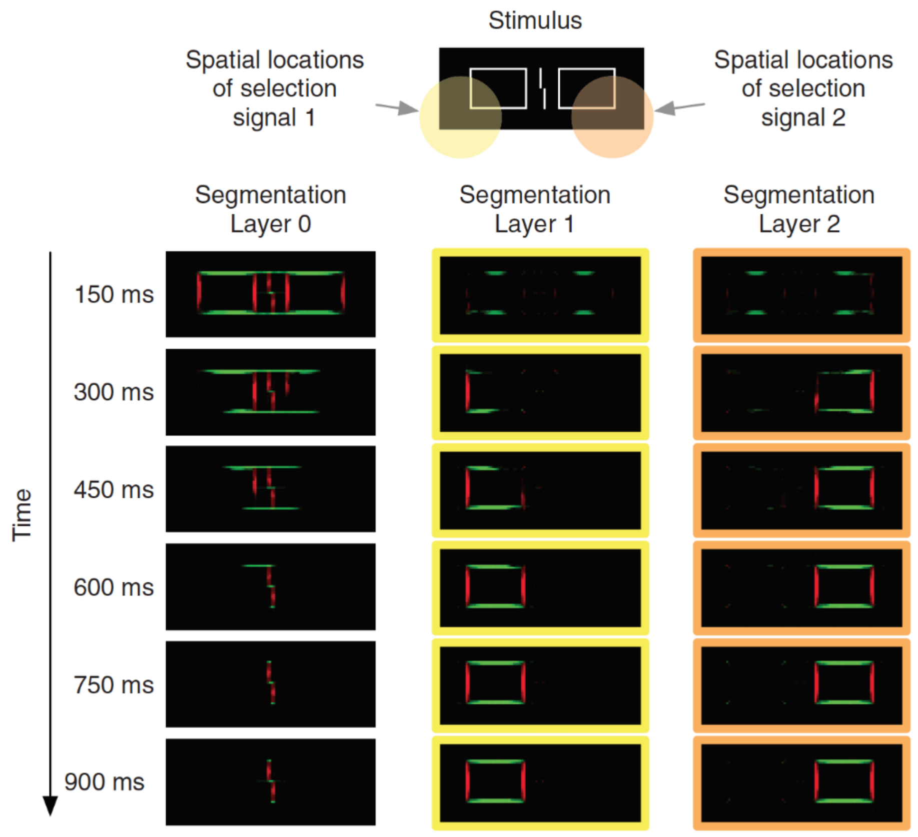


Figure 6. The selection signals are represented by the yellow and orange circles. The signal of the yellow selection signal covers activated neurons, i.e., those comprising part of the bottom and left side of the left box. Over time, this signal spreads to neurons connected to the neurons, i.e., those comprising the rest of the left box. The neurons 'picked out' by each selection signal go to separate segmentation layers, which represent groups of stimulus elements. (Image from Francis et al., 2017, p. 494.)

2.3.2 Modifications of LAMINART

The initial simulations were carried out using a version of LAMINART that is largely the same as that used by Francis et al. (2017) except for some notable modifications, which are indicated in this subsection. This modified version of LAMINART was used to run these simulations to ascertain the viability of a new grouping mechanism.

Francis has eliminated the V2 pooling neurons Layer 2/3. These pooling neurons have been replaced by using local competition with neighboring neurons to determine the strength of the connections between Layer 2/3 neurons.

Additionally, and more importantly for our project, Francis along with one of our group members (M.K.) have been working towards developing a more substantial grouping mechanism for LAMINART. Francis has recently incorporated a Group Damping Cell. When the Group Damping Cell is delayed, a current is generated in 4 possible orientations simultaneously. So, if we set it a global current to be generated in the horizontal orientation while the Group Damping Cell is delayed, all neurons will have increased activation in the horizontal orientation. This will cause, e.g., those comprising the green lines in Fig. 1 to have increased activation and, thereby, become very intense. Currently, the Group Damping Cell is delayed by the person running LAMINART: you set the orientation(s) of desired global increased activation and the amount of time for which you want the Group Damping Cell to be delayed. It is envisioned that this Group Damping Cell delay facilitates grouping among elements of a scene by extending their real and illusory boundaries. And, we speculate that the observer, perhaps with practice, learns to control how this mechanism operates, that is, the observer controls how long it is active and which orientation(s) it strengthens. Currently, however, how to set the direction and amount of time for which the Group Damping Cell is delayed is on a trial and error basis.

Regarding subtask 3A in Fig.2, although we initially thought that we would have to vary this parameter for each rivets image, we learned by running simulations of rivet images that a fixed damping delay of 40ms works well for our purposes of rivet alignment and damage detection. Similarly, found that a selection signal size of 10 pixels worked well.

We also had to modify the manner in which the position of the selection signals were determined, leading us to write a program for subtask 3B and incorporating into LAMINART (subtask 3C). The original version of LAMINART is tailored to computer generated images exemplified in Fig. 7 below, panel D. Plus, these images feature a central target. In contrast, our pre-processed images of rivets have more noise and do not feature a central target. Thus, given that humans use saliency in visual search tasks, we constructed an intensity filter. This filter works as follows. It first converts an image to grayscale. Then it blurs the image using convolution. Next, it divides the image up into a user-specified *m x n* grid. This serves as a high pass filter: in the first pass, the pixels of highest intensity within each grid cell are summed. If there any cell(s) that are above a user-specified threshold, then that location is noted. If there are not any such cells, a second pass is conducted. This is the same as the first pass except pixels of the second highest intensity are also added. Using this program, we can detect the region or regions with the highest intensity, where the number of desired regions is set by the user. This location is used to specify the center of a selection signal in LAMINART.

2.3.3 Simulations conducted

Simulations were run of two psychophysical experiments. These are Experiment 2 carried out by Palmer and Beck (2007) and Experiment 1A (Subjective Ratings) of Manassi, Sayim, and Herzog (2012). These experiments were chosen because they focus on grouping tasks and their stimuli resemble rivet rows and damage, e.g., the lines of the Herzog et al. study may be regarded as idealized smoking rivets. Additionally, although it features crowding experiments, the Herzog et al. study was chosen because we still want to maintain LAMINART's explanatory power for such experiments even with the modifications we will make. We have written a Python program to draw similar images pixel-by-pixel for the purpose of simulating these studies (see Fig. 7 for the stimuli used in each study and those that we created to simulate the studies).

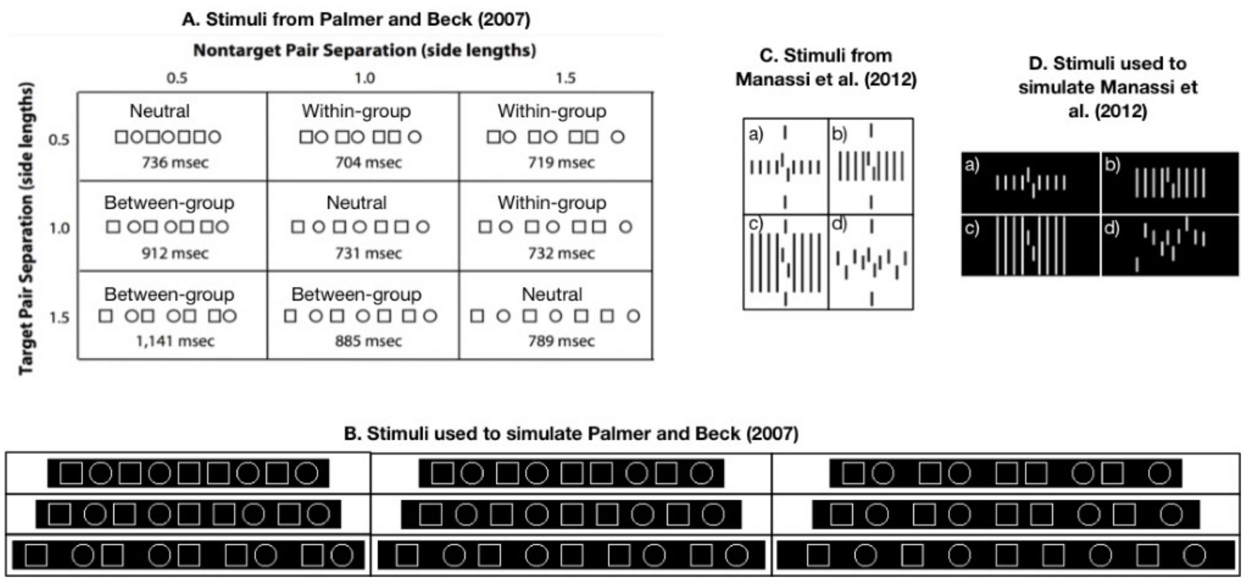
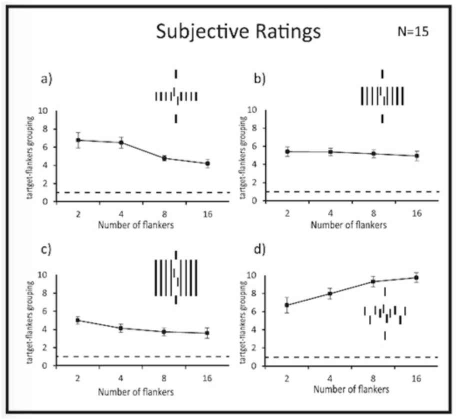

A. Stimuli from Palmer and Beck (2007)
Nontarget Pair Separation (side lengths)
0.5
1.0
1.5
Target Pair Separation (side lengths)
0.5
1.0
1.5
Neutral
736 msec
Within-group
704 msec
Within-group
719 msec
Between-group
912 msec
Neutral
731 msec
Within-group
732 msec
Between-group
1,141 msec
Between-group
885 msec
Neutral
789 msec
C. Stimuli from Manassi et al. (2012)
a)
b)
c)
d)
D. Stimuli used to simulate Manassi et al. (2012)
a)
b)
c)
d)
B. Stimuli used to simulate Palmer and Beck (2007)

Figure 7. Panel A: This figure is adapted from Palmer and Beck (2007, p. 71). To save space they only show 7 shapes per condition, although the actual experiment had 9. There are three main conditions: neutral (spacing is equal between each shape at the length of 0.5, 1.0 or 1.5 times the length of a square), between-group (in which the target pair, which is adjacent squares in this case, are relatively closer to each other), and within-group (in which the target pair are relatively far apart). Panel B: Examples of stimuli we created to simulate the Palmer and Beck experiment. We ran simulations in which the target pair was adjacent squares (pictured) and simulations in which the target pair was adjacent circles (not pictured). Panel C: Stimuli from the Manassi et al. Experiment. Pictured is each condition where there is a total of 8 flankers. However, for each condition Manassi et al. also had stimuli with 2, 4 and 16 flankers. We also created and ran such stimuli. Panel D.: Examples of our stimuli for the Manassi stimuli with 8 flankers. Note that we chose to omit the bars at the top and bottom of the display since they were there to reduce uncertainty regarding the target position.

In the Palmer and Beck experiment, 10 participants were asked to look for a pair of adjacent squares or a pair of adjacent circles in an image. These pairs are the targets. The target pair could appear in positions 3-4, 4-5, 5-6, or 6-7, counting left from right. Participants were asked to rate how easy it is to see that the row contains a pair of adjacent circles or squares on a scale from 1(very difficult) to 10 (very easy). Note that we only simulated stimuli in which the targets were in position 5-6, as depicted in Fig.7, Panel B. The main results from Palmer and Beck's experiment are given in Fig.8, right panel.

In the Manassi et al. experiment, participants were shown the four types of flanker patterns depicted in Fig. 7, Panel C. There could be a total of 2, 4, 8 or 16 flankers, for each condition. In condition (d), the targets are jittered along the vertical position. Note that to simulate this condition, we used a random number generator to create 5 stimuli with 16 flankers. Each of these was manually cropped to versions with 2, 4, and 8 flankers. In the Manassi experiment, 15 participants were asked to judge how much the vernier (i.e., that target in the center that consists of two offset lines) stands out from the flankers on a scale from 0 (vernier does not stand out) to 10 (vernier highly stands out). In the plot of their results (Fig. 8, left panel), Manassi et al. subtracted these ratings from 11. So, high values in the plots below indicate strong grouping between the target and flankers.



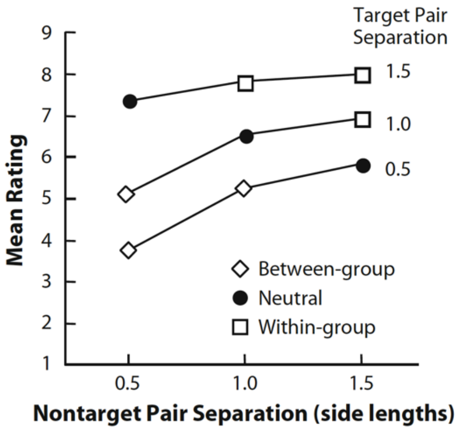


Figure 8. Left: The results from the Manassi et al. experiment for each condition. Right: The results from the Palmer and Beck (2007) experiment.

Our simulations of these experiments were conducted using a truncated version of the most recent version of LAMINART. The system for filling in boundaries with luminance has been omitted. This was done in order to increase the speed of running the model and because the BCS and FCS are largely independent in LAMINART. Although trials were run with alternative and combinations of orientations, the results below are for trials during which the global activation during Group Damping Cell delay was in the horizontal direction only. This is justified by the fact that the stimuli feature rows of elements arranged horizontally from left-to-right and, thereby, the grouping of objects is more likely to be facilitated by increased horizonal activation. For each stimulus, one trial was run for each of the following Group Damping Cell delays: 0, 20, 40, 60 and 70ms. Each trial was run for 20 timesteps, with each timestep being 50ms. This allowed us to see that ranges of values for which elements in the stimuli: did not group at all, had 'good' grouping, or all grouped together. Examples of each condition in Fig.9.

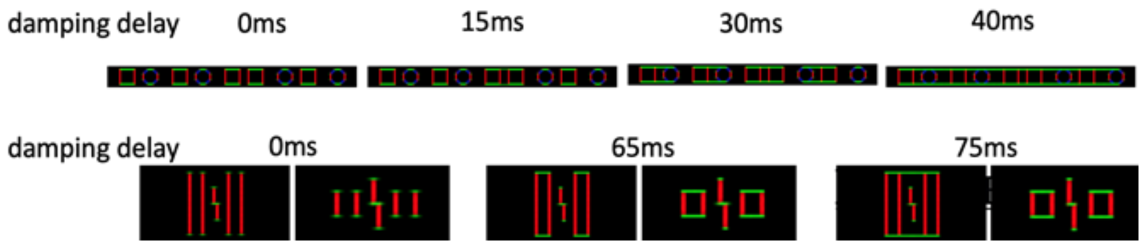


Figure 9. The left images exhibit no grouping. Here the damping delay parameter is set to zero. For high damping delays values, all or nearly all elements group, as depicted in images on the far right. Images in the middle depict output for middling damping delay values. Some of these result in 'good' grouping, i.e., output in which the target, e.g., the adjacent squares or central vernier, does not group with other elements.

Then a total of 30 trials were run for each grouping condition, i.e., no grouping, good grouping and all grouping. For the Manassi stimuli (Fig. 9, bottom row) the selection signals were generated by a random number generator using two normal distributions. Both had a standard deviation of 10 pixels and one was centered at ¼ of the total number of columns of pixels, while the other was centered at ¾ of the total number of columns. These locations were chosen since these are likely locations in which a participant would gaze given the central location of the target. For the Palmer and Beck stimuli (Fig.9, top row), we used our version of LAMINART that we also used for rivet processing, that is, the version with the integrated intensity filter.

2.4 Image post-processing

We had planned to create and implement Python code that uses output from LAMINART to perform two tasks.

First, the resulting program was intended to provide a means of detecting that there is rivet damage and classify the type of rivet damage.

Second, we planned to use the resulting program to create a virtual map of a row of rivets on an aircraft wing that indicates rivets' relative position, rivet damage, and the type of damage.

Although we did not complete this task, we did code that accomplishes some portions of these tasks, which correspond to subtasks 4A, 4B, and 4C, in Fig.2 above.

> 4A. We have a program that overlays the final image from each layer in LAMINART's output onto a pre-processed image. The aim of this is to provide a visual indication as to the extent to which rivets in rows are lined up. Plus, this would serve as input into a stitching algorithm which would allow us to reconstruct the entire rivet row using individual output images.
>
> 4B. We created two error detection algorithms. One works by computing the mean squared error between LAMINART output given a pre-processed image and LAMINART output of an 'ideal' image. This 'ideal' image is one in which there are no damaged rivets, but the locations of the rivets matches those of in the original image. The other allows us to detect damaged rivets based on the structural similarity index. Each rivet is compared against its 'ideal' counterpart and based on the index obtained the rivet is flagged as damaged. The threshold used to decide whether to flag a rivet as damaged or not was obtained by subtracting two standard deviations of the mean of the structural similarity index gathered with test data. However, it currently only works on pre-processed images.
>
> 4C. We created a program to stitch together two images with overlapping parts with the aim of putting the images of sections of rivet rows back together.

2.5 GUI

A GUI was developed that combines different parts of the project including pre-processing, damage detection and LAMINART runs. It also provides the capability for viewing the results of each of those steps in detail.

3. Results and Analysis

We present our results in subsections that correspond to the main modules of Fig.2.

3.1 Robot and raw data collection

The final iteration of the robot was capable of following a line of electrical tape placed upon the wing of an aircraft. This method keeps the vehicle safe during data collection and testing of the other systems but does not allow for autonomous data collection over the entire aircraft wing. In order to accomplish the second goal, we began working on a different navigation style. We decided that we would attempt to create a system similar to those that exist with quadcopter navigation. These systems allow vehicles to find objects that they recognize and navigate in a particular manner once these objects have been found. We decided that we would create a simple image classifier using Keras and train the network on damaged and undamaged rivets. It became quickly apparent that images of damaged rivets are not readily available, and a group member began manually collecting images of damaged rivets. This has led to a fairly inaccurate model, because of this we have not implemented vehicle navigation based on this system.

3.2 Image pre-processing

Pre-processing performs several different operations and transformations on the raw data to get the desired results. For some stages more than one approach was tested, with different results obtained. However, these distinct methods were conserved in the class as none is perfect and each has its advantages and disadvantages. The results obtained from each step of pre-processing can be seen in the following figure.

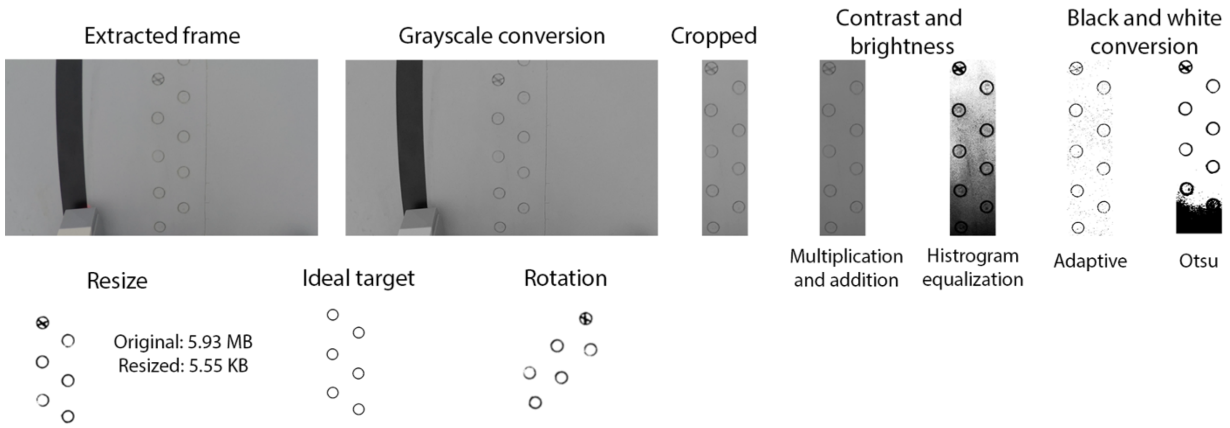


Figure 10. Output from all stages in the pre-processing algorithm.

The extraction of frames and grayscale conversion was straightforward and did not introduce any complications.

On the other hand, cropping the image to focus only in the location of the rivets required the design of a custom method. The circular geometry of the rivets was exploited using Hough transform to detect circles. However, Hough transform requires the right parameters to be able to detect the circles without giving errors, false detections, or missing the detection of a rivet. Through a series of tests, the adequate parameters were obtained, and every rivet is detected, if they can be seen. With the detected circles and "ideal" image is created and used in post-processing to detect damaged rivets.

Contrast and brightness enhancing is done to improve the results obtained in the following stage, black and white conversion. By modifying the contrast, it is intended to highlight the rivets so that the successfulness of the black and white threshold increases. Histogram equalization was tested and although rivets were highlighted, so was the shadow that was cast by the robot, as can be seen in Fig.10. Instead of histogram equalization a multiplication and addition approach was implemented, and it was found that a gain of 0.8 with a bias of 10 improved results.

For black and white conversion two different algorithms were tried. There is no clear better choice between these two, as each has its own advantages and disadvantages in producing results. Adaptive thresholding creates a black and white image where the shadow is avoided but produces noise, represented by black dots scattered in the frame. Conversely, Otsu's thresholding enhances the shadow and ultimately covers some of the rivets completely. However, there is no noise and the rest of the frame is clear and useful for LAMINART. It was decided to crop the portion containing the shadow and use the rest of the frame to continue the process.

Resizing was done with an area interpolation algorithm and a different number of sizes were tested to find the balance between size and definition to keep computation times in LAMINART small while preserving most of the characteristics of the images.

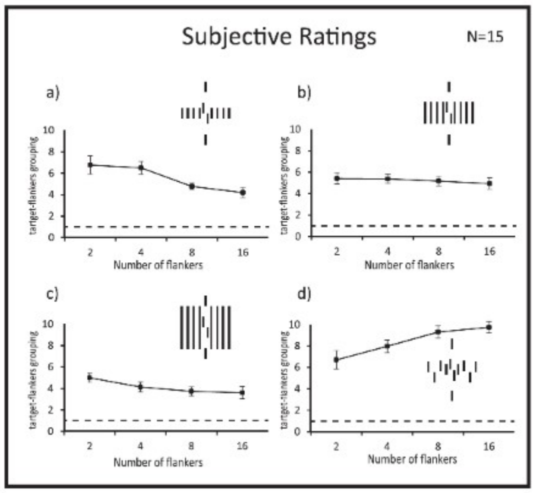
Finally, the rotation needed to have rivets of rows in 45-degree angle was obtained by doing a linear regression of the position of the detected circles by the Hough transform. It is worth noting that the rivets are not in a clear 90-degree angle in every frame captured by the video, and there are many variations on this angle due to the nature of the way the robot moves. Therefore, the successfulness rate of the rotation relies heavily on the detection of the circles. If the circles are detected correctly, the algorithm rotates the frames to a 45-degree orientation no matter what the orientation of the row of rivets in the extracted frame was. The angles of several result images were checked and found to be within variations of 2 or 3 degrees at maximum from the ideal 45.

3.3 LAMINART

Simulations of psychophysical experiments

The results of our simulations of the Manassi et al. and Palmer and Beck experiments are as follows.

The Manassi experiment simulations do relatively well at qualitatively reproducing the subjective ratings of participants. The results are depicted in the right panel of Fig.11.



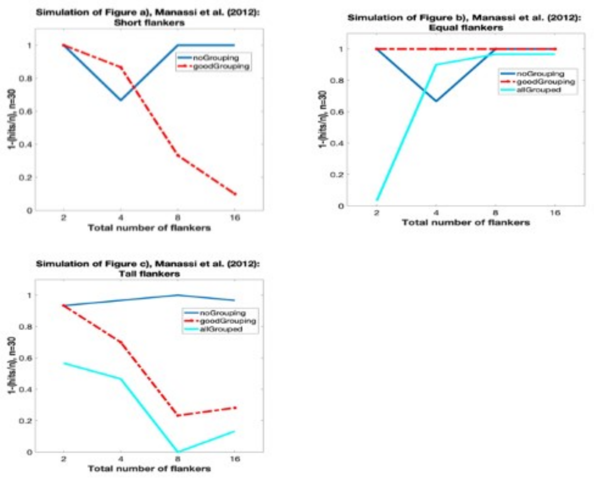


Figure 11. The three plots on the right depict the results of our simulations of the Manassi experiment. Here the y-axis is one minus the number of trials with hits, i.e., trials in which the target ended up in its own segmentation layer, divided by the total number of trials. To facilitate comparison, the results of Manassi are provided on the left.

The slopes of the simulated experiment for the ‘good’ grouping condition qualitatively matches the those of the subjective ratings from the Manassi experiment. This indicates that humans may tune the spread of illusory boundaries in an image such as to optimize the likelihood that they are able to detect a target. Thus, this suggests that LAMINART’s damping delay parameter may play a role in perceptual grouping. However, simulations of further experiments are required to support this claim.

The simulations for the Palmer and Beck study, however, suggest that our intensity filter is inadequate for conducting simulations of psychophysical experiments. Recall that for these simulations, unlike for the Manassi simulations, we used the version of LAMINART with the intensity filter. However, the selection signals seldom went to the location of the target pair of shapes. We believe this is due to the fact that the stimuli used was computer generated and, thereby, relatively homogeneous. Thus, we speculate that either parameters in the intensity filter need to be modified, e.g., eliminate the blurring stage, lower the threshold of the intensity filter, or we need an alternative way to determine the locations of the selection signals.

Another result from our simulations of the Palmer and Beck study is that pairs of adjacent circles are more prone to group compared to pairs of adjacent squares. Stimuli with adjacent circles were more likely to have connecting illusory contours than squares at the same damping delay for the same condition. Such a difference, however, was not reported in the Palmer and Beck study. This could be accounted for by the fact that the study was not designed to test for such a difference. Thus, we have a testable prediction from this simulation, which requires running a behavioral study on human participants to substantiate whether circle targets are less discernable than square targets.

Running pre-processed images

Hundreds of simulations were run in LAMINART using a variety of pre-processed images. For each image, the following parameters were varied: damping delays (which were done in increments of either 10 or 20ms, ranging from 40 to 200ms), orientations of spread during damping delays (which were integers ranging from 1 to 8), and the number of orientations in which each neuron could be oriented (4 or 8 directions). Cropping, adding contrast, rotating and darkening images was initially added to pre-processed images manually. These images were also run in LAMINART and the output was shared with the group in order to give informative feedback to the subgroups working on data acquisition and pre-processing. The aim here was to develop our data acquisition techniques and pre-processing program such that it could output images that can be run in LAMINART and produce grouping among rivets. We have successfully achieved this aim as is demonstrated by the example output in Fig.12.

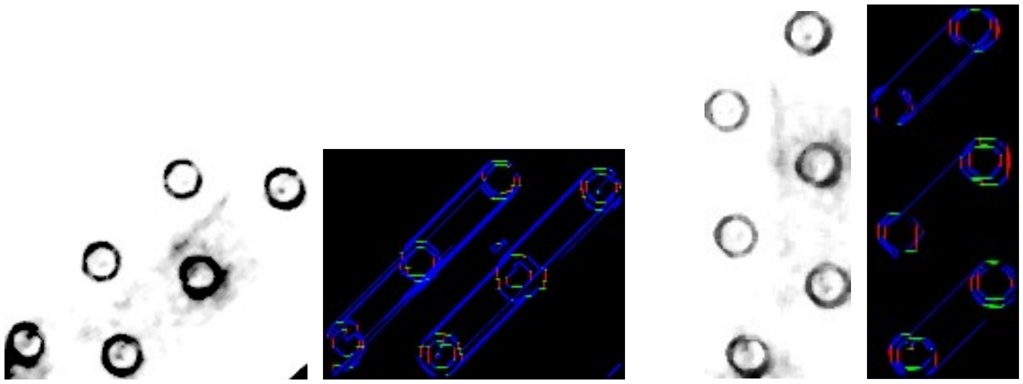


Figure 12. The grayscale images are output from the pre-processing algorithm. The images to their respective right are the output from LAMINART for damping delays of 80ms and 120ms, respectively.

As is evident, given our current data collection and preprocessing techniques we can use LAMINART to draw illusory boundaries both intra-row and inter-row.

After honing the data collection and pre-processing techniques such that we could generate suitable input to LAMINART, we integrated the intensity filter, which allowed us to determine where to put the selection signals based on areas of relatively higher intensity, an example of which is depicted in Fig.13.

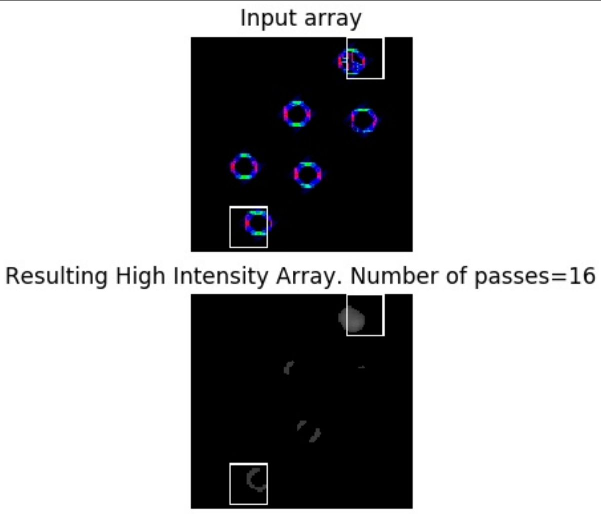
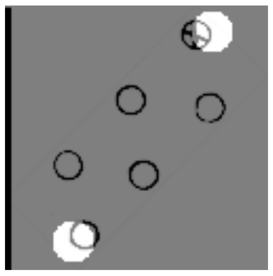
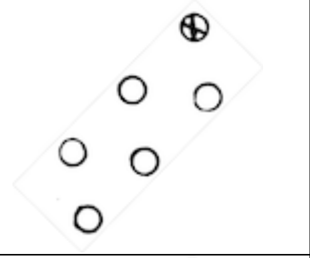
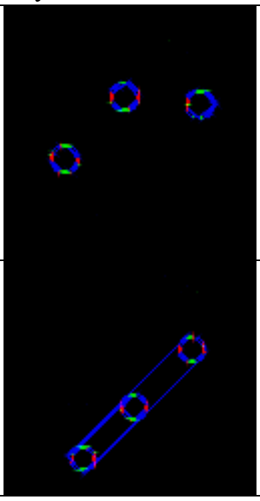
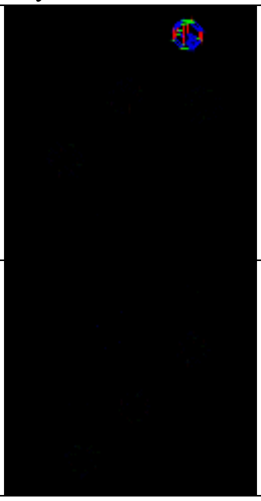
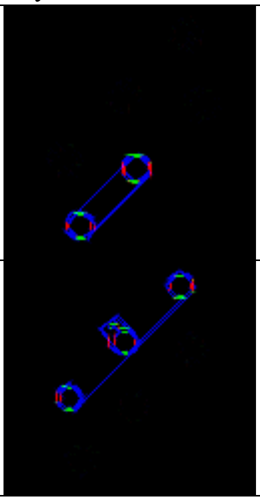
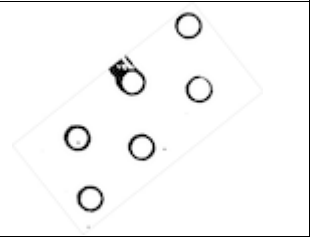
Intensity Filter Input and Results	Resulting Selection Signal Locations
	

Figure 13. The intensity filter finds the most intense regions of the images by converting the image to grayscale (left column, bottom image). It first considers pixels with highest intensity. If two of these regions have a total intensity that is above a user-specified threshold, it stops. Otherwise, it makes another 'pass', i.e., it adds pixels with the second highest intensity and repeats the process until the two highest intensity regions are found.

Then, we ran pre-processed images that included simulated rivet damage. This was damage drawn onto the wing of the CRJ from which we collected our original raw data. Representative output is given in the table below.

	LAMINART Output: Segmentation Layers		
Input	Layer 0	Layer 1	Layer 2
			
			

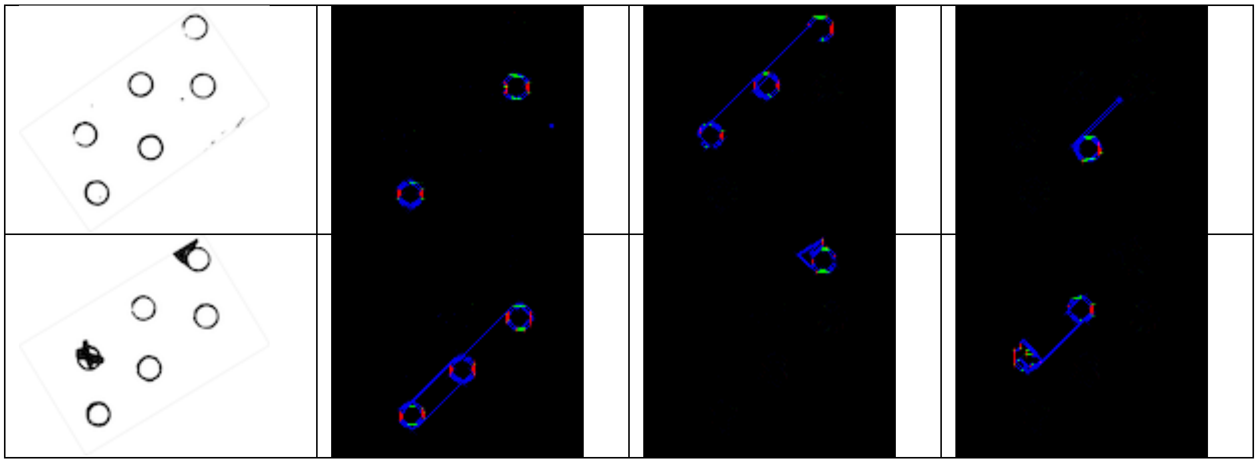


Figure 14. Output of LAMINART with the intensity filter given input of damaged and misaligned rivets.

Clearly, LAMINART allows us to map the relative positions of rivets in a row. We can see whether a rivet is inline or out of line relative to its neighboring rivets, e.g., Fig.14, third row. Additionally, for some types of rivet damage, e.g., the cracked rivet in the first row is segmented into a separate layer, the smoking rivet in the fourth row is similarly in its own segmentation layer. However, some of the damaged rivets, e.g., those in rows 2 and 4, are not segmented and, instead, group with other rivets.

3.4 Image post-processing

As indicated above, although we planned to create and implement Python code that uses output from LAMINART to create a post-processing system, we only completed some of its subsystems. Specifically, we were able to:

4A. Create a program that overlays the final image from each layer in LAMINART's output onto a pre-processed image. As is clear from Fig.15 below, doing so allows us to see the extent to which rivets in each row are aligned. This is indicated by whether the blue illusory contours form between the rivets.

4B. Create two error detection algorithms. One works by computing the mean squared error between LAMINART output given a pre-processed image and LAMINART output of an 'ideal' image. Recall that this image has perfect circles in the same locations as the actual pre-processed image. The other algorithm allows us to detect damaged rivets based on the structural similarity index. As shown in the 'Cracked Rivets' column of Fig.15, it flags a damaged rived with a box. However, it currently only works on pre-processed images.

4C. We created a program to stitch together images with overlapping components, e.g., our rivet images. Although this works well for detailed images, e.g., those taken of a landscape, we have not gotten it to work for rivet images. We suspect that this is due each image's their relatively homogenous appearance.

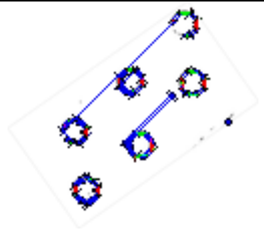
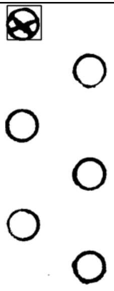
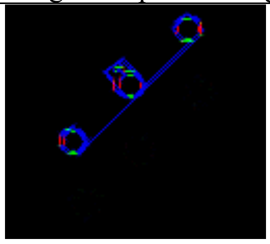
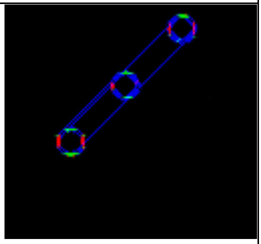
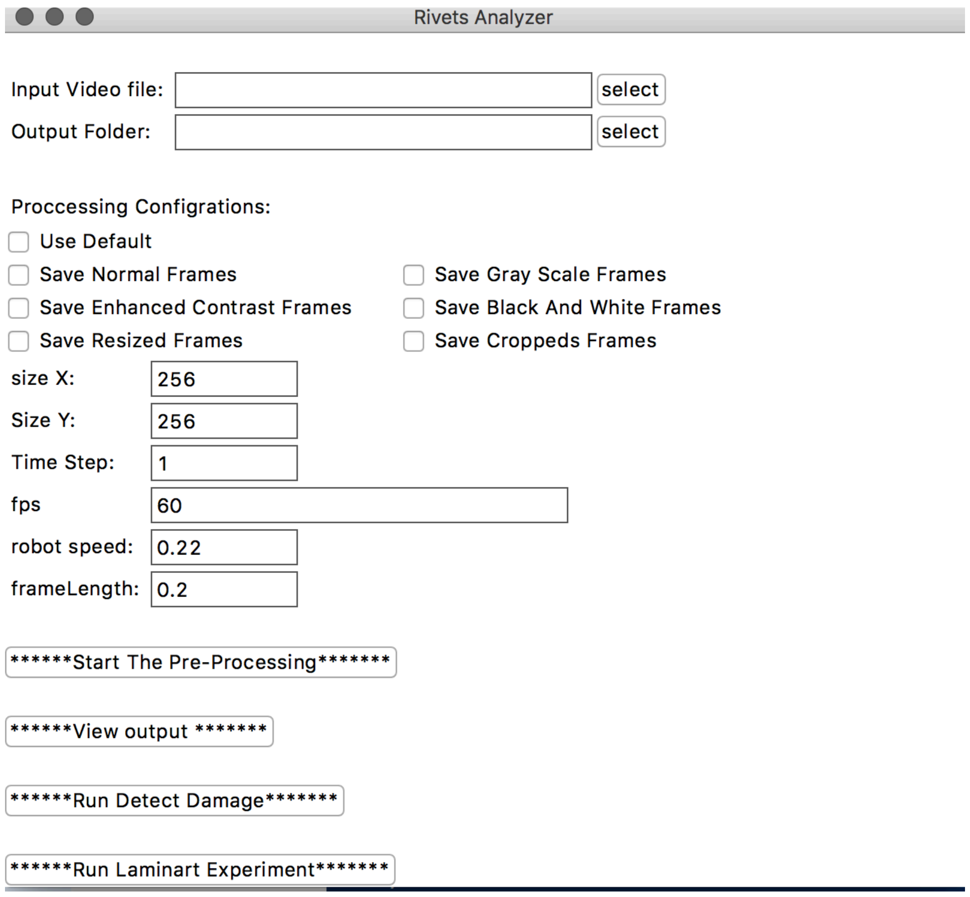
Damage Type	**Misaligned Rivets**	**Cracked Rivets**	**Smoking Rivets**	
Means of Damage Detection	Visual via lack of illusory boundaries among rivet rows.	Structural similarity index, with damage flagged by square.	Mean squared error calculation is relatively high for these two images (23.65), yet lower when there are no smoking rivets (approx. 10). LAMINART output given an 'ideal' image as input is on right.	
Example image(s)				

Figure 15. Table of types of rivet damage and how we propose to detect them.

3.5 GUI

A graphical user interface has been built to combine different parts of the project (pre-processing, damage detection and LAMINART stimulation) to one point of execution. The figure below shows the main interface of the GUI. The user specifies a video file to process and the location where the results should be stored.

The preprocessing algorithm has six different stages that can be viewed separately. The decision is left to the user to decide what outputs to be generated. Parameters that control the pre-processing can be configured through the interface as shown in the figure below.


Rivets Analyzer
select
select
******Start The Pre-Processing*******
******View output *******
******Run Detect Damage*******
******Run Laminart Experiment*******

Input Video file:
Output Folder:

Proccessing Configrations:

- [ ] Use Default
- [ ] Save Normal Frames
- [ ] Save Gray Scale Frames
- [ ] Save Enhanced Contrast Frames
- [ ] Save Black And White Frames
- [ ] Save Resized Frames
- [ ] Save Croppeds Frames

size X:	256
Size Y:	256
Time Step:	1
fps	60
robot speed:	0.22
frameLength:	0.2

Figure 16. The main interface of the GUI.

There are four main actions that can be executed through the interface:

1. Start The Pre-Processing: Process the input video to generate frames that are ready to be processed by LAMINART and the damage detection algorithm.
2. Run Detect Damage: Runs the damage detection algorithm for each frame that is generated and store the results.
3. Run LAMINART : Runs LAMINART experiments for each frame and store the results.
4. View output: The user can see all the outputs related to a given frame through this interface, the figure below shows a snapshot of the output viewer for frame 0.

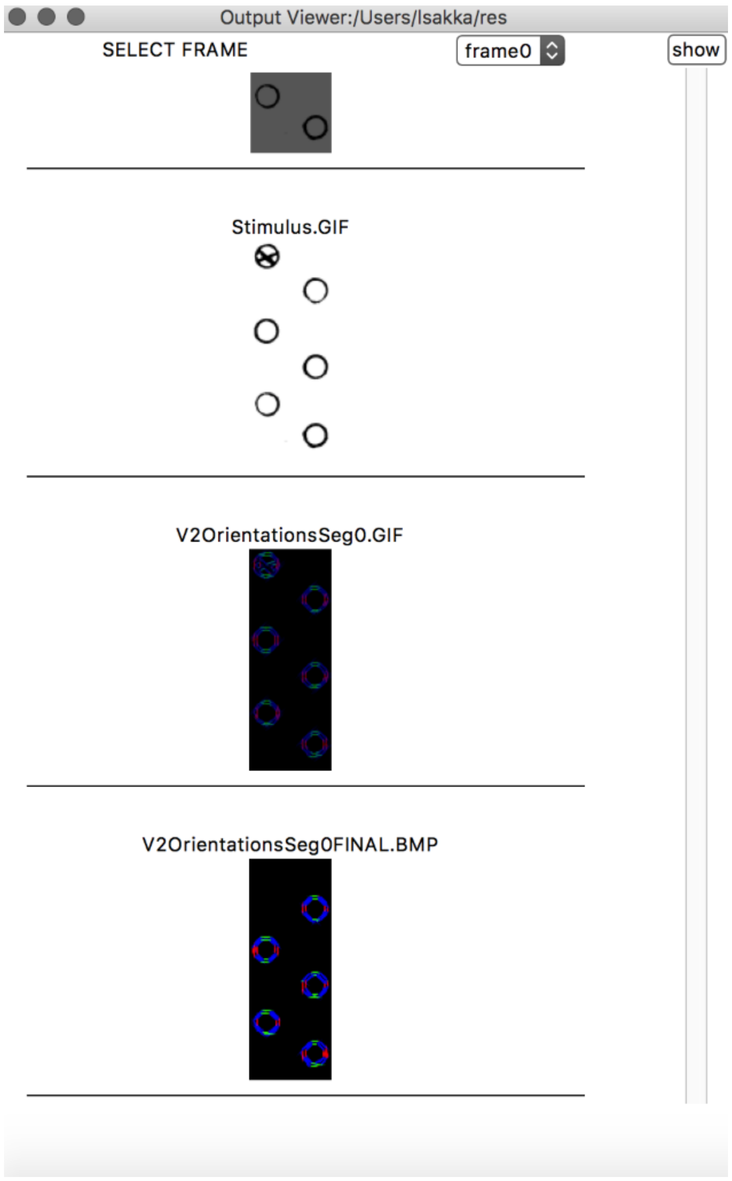


Figure 17. GUI Output viewer.

4. Discussion

Again, we organize this discussion in terms of each main module in Fig.2.

4.1 Discussion of robot and raw data collection

The data collection platform can collect video on relatively flat surfaces of an aircraft but did require the operator to pre-define the operating area and monitor the vehicle. This set up was not ideal, but it was sufficient for testing. As stated earlier, progress was made on creating an autonomous navigation system, but was unfortunately unable to be completed.

Data was gathered of both damaged and undamaged rivets, shown in Fig.18. This data was used for training the model of rivets, and for testing the grouping algorithm. Our focus was on cracked and smoking rivets. This was the damage that was simulated (Fig.19, left panel), but we were also able to find examples of crushed rivets, shown in Fig.19, right panel.

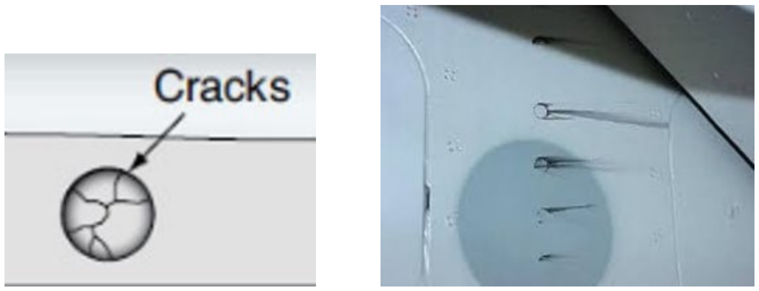


Figure 18. Left panel: A depiction of a cracked rivet. Right panel: Examples of smoking rivets.

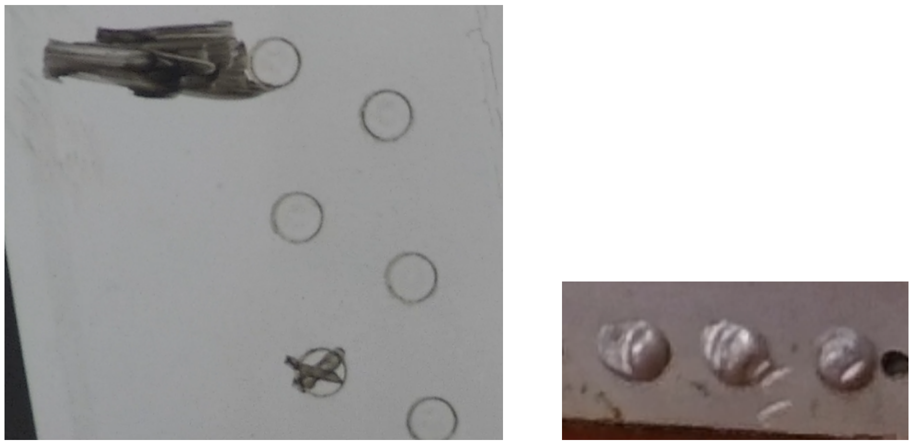


Figure 19. Left panel top: Simulated smoking rivet. Left panel bottom: Simulated cracked rivet. Right panel: Examples of smoking rivets.

4.2 Discussion of image pre-processing

Image pre-processing managed to produce output with the adequate specifications needed for LAMINART, i.e. black and white images that are very small in size and have rows of rivets in a 45-degree orientation. The algorithm retains the flexibility to adapt parameters to meet changes in the requirements that could arise from the use of different methods to detect damaged rivets. This can be seen in the fact that pre-processing was modified to produce not only input for LAMINART, but rather input for the error detection used in post-processing, which includes both an image with rivets and an ideal target.

Pre-processing also gave valuable insight that was used for data collection. These include lighting, and position and orientation of the camera, that were used to produce several iterations of the robotic platform used to gather video of rivets.

Further work on pre-processing is contingent on new requirements by either LAMINART or other techniques used to detect damaged rivets.

4.3 Discussion of LAMINART

Simulations of psychophysical experiments

Our simulation of the Manassi experiment indicates that properly turning the damping delay parameter is essential for grouping elements of a scene that facilitates the detection of a target, e.g., a damaged rivet. However, the simulations of Palmer and Beck experiment indicate that intensity filter does not provide a suitable means for determining selection signal location. In this simulation, the selection signals seldom went to the location of the target pair of shapes. Further, the intensity filter has an additional issue: it involves several user-specified parameter values, e.g., size of cells in the grid, desired intensity threshold. In turn, this filter seems be a reiteration of the selection signal issues, i.e., its diameter and location are user-specified. A potential way around this that we are exploring is to make a probability distribution based on the illusory boundaries that form at each time step. The location of a selection signal at a time step would be a number sampled from this distribution.

Another result from our simulations of the Palmer and Beck study is that pairs of adjacent circles are more prone to group compared to pairs of adjacent squares. Thus, we have a testable prediction from this simulation, which was not investigated in the original study. This requires running a behavioral study on human participants to substantiate whether circle targets are less discernable than square targets.

Running pre-processed images

LAMINART is extremely useful in mapping the relative positions of rivets in a row: its output provides a visual indication of whether a rivet is inline or out of line relative to its neighboring rivets. Additionally, its segmentation system provides some indication as to whether a rivet is damaged. However, this does not work in all cases. Notably, some smoking rivets as well as some cracked rivets that have their circular boarder intact group with undamaged rivets. However, LAMINART reliably does not group cracked rivets that are smaller than undamaged rivets or have irregular boundaries. Thus, LAMINART is useful for detecting particular types of rivet damage and provides a good way to visually represent the relative positions of rivets in and across rows.

4.4 Discussion of post-processing

Our post-processing system is incomplete. Due to the data collection and pre-processing stages being more time-consuming than anticipated, we did not have adequate time to complete it. Nevertheless, we have made progress on some of its components, such as the error detecting programs and the code to overlay LAMINART output onto other images. So, although these components have not been integrated into a system that provides an informative overlay on an orthomosaic, it does not seem an insurmountable task to complete this system in future work on this project.

4.5 Discussion of the GUI

The current GUI created allows all pre-processing to be completed in a user-friendly way. Users are able to search folders for video files, select processing settings, and choose output folders from a centralized, and easy to understand menu. Orthomosaicing capabilities were to be integrated into the GUI, but this section of the project was unable to be completed.

5. Conclusions

The system that has been created includes: a robotic data collection platform, an image processor, and a perceptual grouping network. These systems working in concert are capable of collecting visual data of aircraft rivets, separating frames from the video, preparing these frames for LAMINART, applying perceptual grouping techniques to analyze groups of rivets with LAMINART, and a GUI that allows control over data processing and LAMINART. This grouping map provides us with a virtual map of undamaged rows of rivets on the aircraft. A learning algorithm was developed to determine if rivets were damaged, but was unable to be adequately trained to be reliable. A photo stitching program began development as a way to begin creating orthomosaic images and models from the data collected, but was unable to process the data sets provided. The current system provides a strong proof of concept for this style of inspection, and with additional work could be turned into a powerful inspection tool.

6. Acknowledgments

We thank Professor Ropp for providing us with the EV3 robot and access to the CRJ.

7. References

Chang, H., Grossberg, S, & Cao, Y. (2014). Where's Waldo? How perceptual, cognitive, and emotional brain processes cooperate during learning to categorize and find desired objects in a cluttered scene. *Frontiers in Integrative Neuroscience, 8*, 1-46.

Dodd, M. D., & Pratt, J. (2005). Allocating visual attention to grouped objects. *European Journal of Cognitive Psychology, 17*, 481-497.

Doerig, A., Bornet, A., Rosenholtz, R., Francis, G., Clarke, A. M., & Herzog, M. H. (2018). Beyond Bouma's window: How to explain global aspects of crowding? *Manuscript under review at PLOS Biology*.

Dresp-Langley, B., Grossberg, S., and Reeves, A. (2017). Perceptual grouping—The state of the art. *Frontiers in Psychology, 8*, 67.

Francis, G., Manassi, M., & Herzog, M. H. (2017). Neural dynamics of grouping and segmentation explain properties of visual crowding. *Psychological Review, 124*, 483-504.

Glaessgen, E., & Stargel, D. (2012). The Digital Twin paradigm for future NASA and U.S. Air Force vehicles. *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials*

Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA (pp. 1-14). Honolulu: AIAA. doi:DOI: 10.2514/6.2012-1818

Grossberg, S. (1999). Link between brain learning, attention and consciousness. *Consciousness and Cognition, 8*, 1-44.

Grossberg, S., & Mingolla, E. (1985). Neural dynamics of perceptual grouping: Textures, boundaries, and emergent structures. *Perception and Psychophysics, 38*, 141-171.

Henriques, M. J., Fonseca, A., Roque, D., lima, J. N., & Marnoto, J. (2014). Assessing the quality of an UAV-based orthomosaic and surface model of a breakwater. *FIG Congress*, (pp. 1-16). Kuala Lumpur, Malaysia.

Henriques, M. J., & Roque, D. (2015). Unmanned aerial vehicles (UAV) as a support to visual inspections of concrete dams. *Second International Dam World Conference* (pp. 1-12). Lisbon, Portugal: Dam World.

Manassi, M., Sayim, B., and Herzog, M. (2012). Grouping, pooling, when bigger is better in visual crowding. *Journal of Vision, 12*, 1-14.

Matthias, L., Martin, S., & Elgar, F. (2004). A ubiquitous computing environment for aircraft maintenance. *Proceedings of the 2004 ACM symposium on Applied computing* (pp. 1586-1592). Nicosia, Cyprus: ACM.

Palmer, S. E., & Beck, D. M. (2007). The repetition discrimination task: An objective method for studying perceptual grouping. *Perception & Psychophysics, 69*, 68-78.

Simpson, E. J., Wooster, J. M., Smith, E. L., Trivedi, M., Vernimmen, R. E., Dedi, R., Shakti, M., & Dinata, Y. (2016). Tropical peatland burn depth and combustion heterogeneity assessed using UAV photogrammetry and airborne LiDAR. *Remote Sensing, 8*(12), 1-21.

Tuegel, E. J. (2012). The Airframe Digital Twin: Some challenges to realization. *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA.* Honolulu: AIAA. doi:DOI: 10.2514/6.2012-1812

Wagemans, J., Elder, J. H., Kubovy, M., Palmer, S. E., Peterson, M. A., Singh, M., & von der Heydt, R. (2012). A century of Gestalt psychology in visual perception I: Perceptual grouping and figure-ground organization. *Psychological Bulletin, 138*, 1172-1217.

Zarco-Tejada, P., Diaz-Varela, R., V.Angileri, & Loudjani, P. (2014). Tree height quantification using very high resolution imagery acquired from an unmanned aerial vehicle (UAV) and automatic 3D photo-reconstruction methods. *European Journal of Agronomy*, 89-99.

Appendix A: Roles of Team Members

TEAM MEMBER	CONTRIBUTIONS
Javier Belmonte	• Image preprocessing: research, coding, and implementation. • Developed an algorithm to extract suitable input for LAMINART. • Improved preprocessing to allow extraction of every step of the algorithm for analysis. • Improved robustness of the algorithm by handling errors. • Improved interactivity of the preprocessing algorithm by giving options and messages to the user. • Developed error detection algorithm. • Wrote meeting logs.
Xin Ding	• Image preprocessing: research, coding, and implementation. • Applied PIL to test images collected by the camera mounted on a drone. • Wrote meeting logs. • Developed GUI to execute the preprocessing algorithm, get results from the user • Developed two ways to show output images continuously (automatically & by clicking "next" button)
Maria Kon	• Found existing psychophysical studies that feature grouping and use stimuli that resemble rivet rows and rivet damage. • Created images for and running simulations of psychophysical experiments using LAMINART. • Ran pre-processed images in LAMINART • Designed the intensity filter for selection signal location. • Created a mean squared error algorithm for LAMINART output. • Created algorithm to overlay LAMINART input onto a pre-processed image. • Sent LAMINART scripts. Wrote instructions on how to install one of its dependencies (NEST simulator) and how to run LAMINART on Linux, Mac, and the Scholar cluster remote desktop.
Laith Sakka	• Robot programming and testing. • Collected raw video data. • Wrote cropping algorithm for raw images. • Developed GUI to include LAMINART and Error Detector • Setup Github for group and instructed group how to use it.
William Weldon	• Found existing aviation studies that focus on remote sensing and aviation maintenance • Robot programming and testing. • Collected of raw video data. • Processed raw data as an orthomosaic. • Created image stitching algorithm. • Created, trained, and gathered training images for an image classifier. • Setup Purr account for group and uploaded video and raw images.

Appendix B: List of Python tools used

In Python we wrote: a program that controls the robot, a pre-processing program, a graphical user interface program, a cropping program, a program can be used to create stimulus images for running simulations of psychophysical experiments using LAMINART, an program that uses image intensity to determine the location of LAMINART's selection signals, error detection algorithms, an rivet classifier, and an algorithm to map LAMINART output to the pre-processed images. Some tools used in these programs are: numpy, sys, os, random, and imageio.

In the truncated version of LAMINART that we used to run the simulations, the following Python tools are used: os, sys, numpy, random, nest, matplotlib, and imageio. 'nest' is the NEST simulator, a tool for creating and running neural networks.

Appendix C: List of Python modules

PYTHON MODULE	PURPOSE
ShapeTypeList()	Gets user input that specifies the type of shapes in the stimulus image from left to right. Returns a list of shape names, e.g., [‘circle’, ‘square’].
SpacingList()	Gets user input that specifies the spacing in pixels to the right of each shape. Returns the total number of pixel columns that the image requires and a list of the number of pixels to the right of each shape.
OpenCV	Process video to extract images. Conditions these images by contrast enhancing black and white conversion and resizing to produce suitable input for LAMINART.
time	Used to compute duration of preprocessing computation.
collections	Needed to create ordered dictionaries.
Os and pathlib	Modules that provide ways of using operating system dependent functionality, such as reading or writing files and paths manipulations.
rgd2gray()	Converts a colored image from LAMINART output into a grayscale image.
IntensityMaxFinder()	Given an image, finds the intensity value of the brightest pixel, plots all pixel intensities, and writes an image that has white pixels where the brightest pixels are.
IntenseAreaFinder()	Given an image, finds the most intense areas and produces an image that encircles these areas.
Mse()	Computes the mean squared error given two images.
tkinter()	Build graphical user interface with message checkbox, button, and text input.
PIL	Transfer image type into the type GUI can execute.