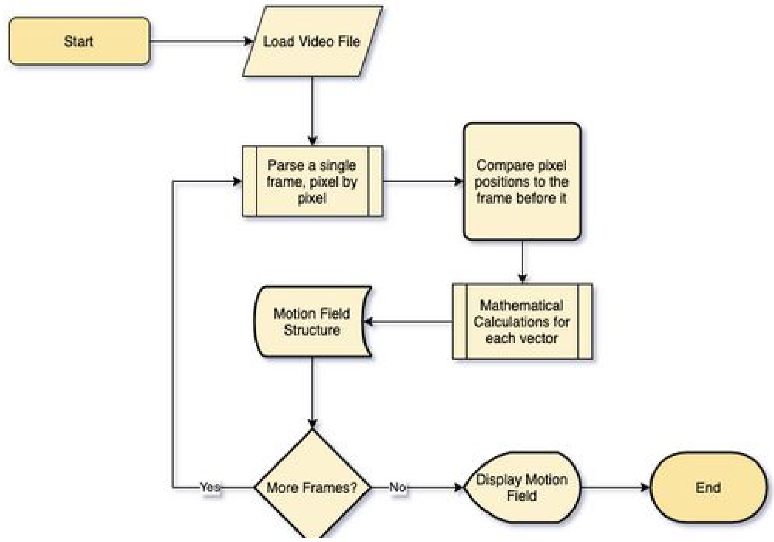
Jair Gonzalez
Rahul Singh
CSC 664: Multimedia Systems
5/16/2021

Final Project: Determining motions fields using Optic Flow

For the average human, determining motion can be simplified into visual tracking of objects moving across a scene. A person can then extract the information that they are looking at and then calculate various points to perform tasks such as driving, playing a sport, or even just walking through the park. Being able to extract information from visual cues is second nature for humans and within recent advancements with technology, computers have adopted the capabilities to also determine motion using video processing. More specifically Optic flow, a specific term that could be defined as the pattern of motion relative to an observer and the scene in which motion is being created. For this project, I aim to create a application that uses video processing to determine optic flow on given data from the professor which include a set of zebra fish images in both binary and grey scale, plus a video of the actual zebra fish moving around in there tanks. Being able to create a solution to optic flow can seem like a simple process but over time of research and development, it proved to be quite a difficult task to complete. Although challenging, optic flow is a key technology that brings new capabilities to our daily lives including enabling autonomous vehicles, improving video editing software and even for bio-medical treatments at molecular levels. The following report will outline main ideas and concepts that I have explored to try and create a solution to this problem, I try to cover area's including my own proposal, a high-level description of my idea, the research that I have come across, my solution and finally a reflection on what I have completed as well as the shortcoming of my solution.

I will attempt to create an application that would be able to quantify a set of frames within a video and then use the data from those frames to calculate a what is known as frame differencing. With frame differencing I can detect individual pixels that are displaced within different frames then using various techniques and some proposed methods from my research, I plan to create a application that applies optic flow to a video format. There are many difficulties when working to create an optic flow at a pixel level in computer vision that could give us an inaccurate result of motion from our video such as poor picture quality, pixel skipping and even shadows. The method that I attempt is called the "Lucas-Kanade method" that uses a local approach on identifying specific pixels to track instead of tracking all the pixels, this local approach could end up giving us better results in determining motion fields and could be resistant to many of the problems that general optic flow faces. Main things that I would like to accomplish is being able to track pixels throughout a whole video at a consistent rate.



I have formulated a workflow diagram that highlights a high-level approach on what I plan to do in my application. First off, a side of loading video file is to find a way to parse the individual frames of a video so that I can reach the contents of each individual frame. Next thing would be to extract I would want to be able to access individual pixels within a frame and map them out using matrixes or vectors. Next, I would apply techniques that would allow me to calculate the vectors of pixels, that would, with this I would be able to create a motion field that that could later be displayed as a cue sign of motion. I would attempt to loop this process throughout an entire video file until there are no more frames to parse. My initial thoughts of going with this approach would be to first use the technique of frame differencing, which is a global approach to an entire scene, that uses the X, Y and greyscale value of each individual pixel and then compares those values to the next frame. The underlying result would be those pixels that have a change in greyscale value would be highlighted which would be a sign of motion, and those pixels that kept the same greyscale value would be no motion. Using this basic formulation, I would move onto applying more techniques that would enable me to track the positions of individual pixels.

As an introduction to the subject of what Optic Flow is, I reviewed a Lecture by Dr. Anthony French, Titled “Optic Flow”, Dr. French. The lecture goes through the idea of optic flow by defining it as “understanding what is moving in a image, at the pixel level… how things are moving across a whole image” (4), then follows it up with use cases for this technology in image stabilization, drone navigation, and autonomous vehicles. Within this lecture, there are key assumptions that Dr, French mentions that must be made when calculating optic flow including “lighting does not change” (4), what this means is that grey scale values are relatively consistent within a scene. This assumption must be made because inconsistent objects such as shadows could be picked up from using an optic flow solution. With many helpful illustrations, the next topic that Dr. French covers is the Optic Flow Equation, which is the

$$I(x,y,t)=I(x+dx,y+dy,t+dt)$$

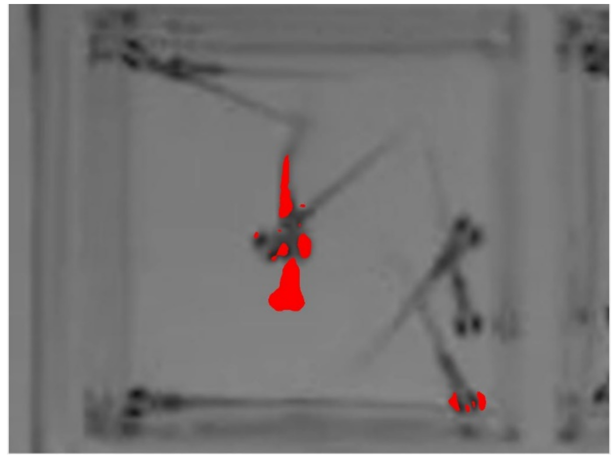
Intensity of a grey scale value (I) within a scene coordinates (X & Y) different frames, but we call this time (T). When extracting this information from a frame, we can calculate the derivatives form that same grey scale value on where it has moved to within different frames. This then leads us to be able to get the pixels optic flow vector also mathematically called (U, V). When going through all this information in the lecture I thought that this was great information, I am able to now use a equation that I could apply into my calculations on being able to calculate motion, in addition the concerns about shadows being detected in optic flow was something that I wanted to also think about in my solution. I thought of strategizing by trying to create a threshold on how different greyscale values needed to be to activate optic flow in my program.

The next following piece of research that I took interest in was a research paper titled "WormAssay: A Novel Computer Application for Whole-Plate Screening of Macroscopic Parasites" by various researchers including Jiri Gut, K. C. Lim, Rahul Singh, James McKerrow, and Judy Sakanari. Within this article the researchers aim to develop a low-cost video processing system that would be able to determine the efficacy of anti-parasitic drug treatments, specifically on adult worm agents. The researchers would perform screening on 24 plates, each containing *Brugia Malayi* an adult parasitic worm. Then researchers would test various drugs on each plate and then be able to calculate success by measuring the worm's movements using there visual tracking system, but the best results would return a 0 – 2 movement units which would mean that the drug was very effective in killing the parasite. Further down the article we come to how they implemented optical flow for their tracking of movement the plates, the researchers used two algorithms the first one which aimed to "derive(s) the velocity from the optical flow vectors of the luminance component of the video stream from a pair of adjacent frames approximately 100 ms apart"(1) and then implemented a iterative version of the "Lucas-Kanade method", a local approach to optic flow. For the second algorithm, the researchers created a way to detect changes in a group of pixels but it only chooses 5 frames within every second to do so, but instead use a new approach for bringing consensus of pixel change. They use what is called "Consensus Voting Luminance Difference", a scheme where color and a voting system to determine if a pixel's value has changed. My initial thoughts of this research came to me that I should possibly try and implement the "Lucas-Kanade method" approach as well in order to quantify better results in my Optic flow approach, and possibly as a advanced learning objective, I could also look into other voting schemes that could also let me determine if pixels are changing with my optic flow.

A recommend research document proposed to me was Titled "Block-Matching based Optical Flow Estimation with Reduced Search Space Based on Geometric Constraints" by various researchers including Bernd Kitt, Benjamin Ranft and Henning Lategahn, that proposes a new approach to optic flow by using block-matching for the application of Autonomous Vehicles. Within this article, the researchers introduce the inefficiencies within gradient based approaches and matching-based approaches, stating that either the motion vectors are too small or too large to create an effective solution for optic flow. The approach would be to implement "virtual images with predefined epipolar geometry from known vehicle motion"(3). Epipolar Geometry is the relationship between two individual observers of a static scene, essentially two different point of views looking at the same scene. The experiment for these researchers is to use a already captured image along with a image taken in real time to enhance the integrated navigational system. This create a system that allows the researchers to only capture optic flow vectors of real motion being captured within there sensors. I think that this approach is very useful for capturing dense motion scenes, such as in a busy intersection. If there are many agents of motion within a scene optic flow calculation could be difficult to solve without data previously stored.

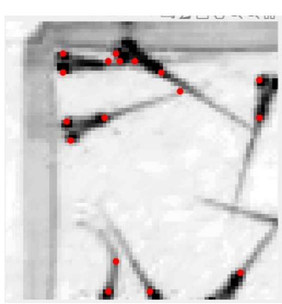
Finally for the last research document that I wanted to focus on was learning about how the lucas-kanade algorithm could be applied into my solution, I looked over a Research document Titled "Optic Flow" by Dr. Florian Raudies, within this paper the main concepts of Optic flow are introduced, and Dr. Raudies brings an important section where he outlines important constraints for the estimation of Optic Flow. He claims that "Most constraints for methods that estimate optic flow can be derived from the continuity equation" (4), the continuity equation includes complex topics such as planar constrains, sampling conditions, bayhesian constaint and even generalized structure tesnors. All important factors when calculating very accurate optic flow vectors. Though out the rest of the document he uncovers many different types of methods, first starting with the method I was most personally interested in which was the Lucas-Kanade method which assumes that all gray scale values changes are between two frames. $f(x+\Delta x,y+\Delta y,t+\Delta t)$ by using this equation and then applying Talor series expansion we are able to calculate velocities also known as our (u,v) mentioned above.

For the work that I have done, I decided to create a design pattern to be singleton. My thoughts were simple, I would want to be able to work with many components into one single system and have it work. Although, my finals files are a conjunction of various

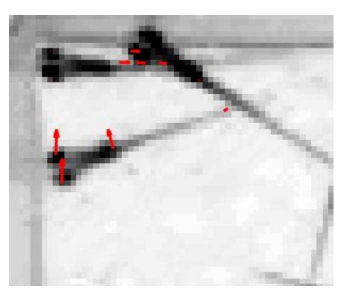


For my experimentation with the project, my first task that I wanted to accomplish was being able to figure out frame differencing using MATLAB, I was able to analyze from information at a frame-by-frame rate by using a tool called video reader. Using this tool, I began to analyze each individual frame by creating a matrix that was the length and width of a frame. With the same tool I extracted the values from each pixel and ran them into a loop differentiating those values that were not the same. For those that had a changed value, I created a layer to highlight those specific pixels in red thus creating a form of frame differencing. This preliminary stepped posed to be somewhat ineffective due to not all motion within the scene being captured. I believe that the results that I am getting is due some way that I am applying frame differencing in my calculations which could be leaving pixels from not being differentiated.

There were some difficulties when extracting the individual contents from each frame so I deciding to start looking into a library that MATLAB offered which was called the vision toolbox that gives me a new approach on extracting information from frames. Instead of diving into video formats, I first worked on finding the differencing within 2 consecutive frames, when being able to quantify the frames at a consistent rate. I began to look into If I could apply the “Lucas-Kanade method” by applying there algorithm. I first loaded the two images into my program, calculated corners that I would be able to track. Once selecting points that I wanted to plot, I was able to draw out the vector’s from both images.

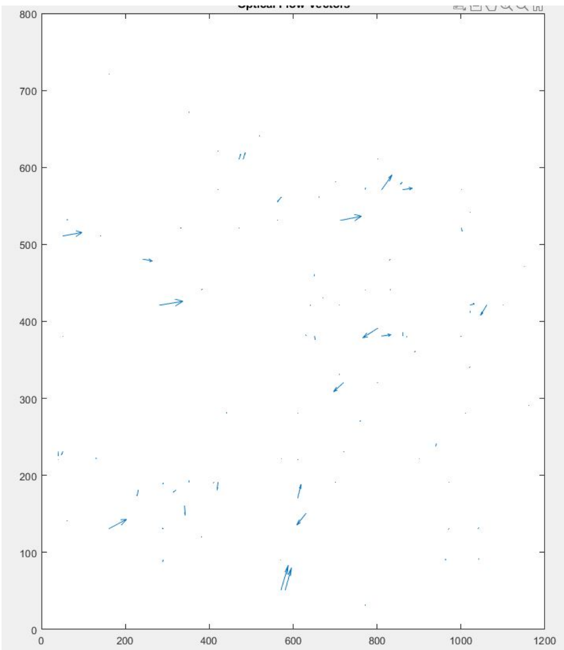


points to plot (image 1)



Point vectors (image 2)

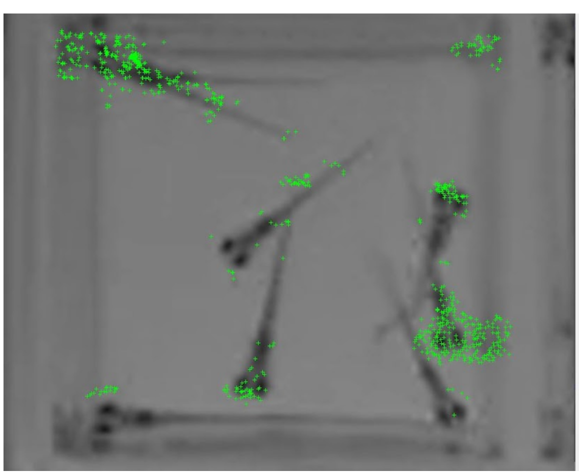
Although that the movement does reflect some sort of vectors, I believe that the method in which I was implemented it was also inefficient. When applying the technique of choosing specific points to track, I was not accounting for noise within my image, this game me some vectors that were inconsistent with the greyscale values when I decided to extract the values from the frames.

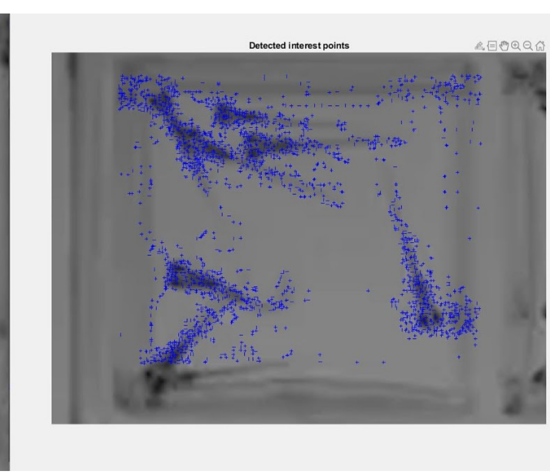


Within MATLAB there exists a Optic Flow function that allows that calculates some of the frame differencing to our given data. The following experimentation would be to apply the function and the techniques from prior to try and extract vectors from the same given data. Using a graph, I was able to then plot those vectors in an illustrated way. I found that the vectors that were being created to be a good quantification of the motion vectors. Some observations that I was seeing was that motion was happening with all area's of the scene, and those vectors were being captured within this experiment.

Though out most of the given data that I was using to create these experiments, I found the data I was using to be rather noisy. Which lead me to believe that some of the inconsistency's that was going on were related to pixel skipping or losing track of specific pixel regions. I decided to do some minor video editing to my data by simply cropping a section of the video to try and get better results. Some of my findings were significant. I found that by reducing the image resolution, I could better quantify some of the results I was getting from previous experiments.

For my final implementation of all previous experiments, I attempted to create a Optical Flow application that uses MATLAB's tools to be able to calculate corners to plot and apply it in a local approach. For the work that I have done, I decided to create a design pattern to be singleton. My thoughts were simple, I would want to be able to work with many components into one single system and have it





work. For this application, I wanted to outline specific points within the video that I wanted to capture.

Once I passed in those values is where I began to implement the lucas kanade method into a video reader function within MATLAB.

Throughout this project I believe I was left with more questions than answers about optic flow. Although I implemented a technique that was already established, I do not believe I went into a depth that was sufficient to cover a ideal solution to optic flow. Although my initial goals were somewhat met, I believe that there is much room for improvements in many area's of my solution such as creating a better system of analyzing frame data. Most of the experiments I completed often were left with inaccuracy's that I was not able to quantify and calculating success was difficult. What I thought to be successful was a analysis of pixel vectors, but being able to actually determine if those vectors were actually accurate was a complete mystery. I think what I found most challenging was various types of mathematical procedures and various algorithms to just understand how optic flow was working.

Things that I did learn that I found valuable included processing video and image files, a procedure that I have never done in previous course work as well as using MATLAB programming with its various tools to be able to create an application.

Bibliography

(1)C. Marcellino, J. Gut, K. C. Lim, R. Singh, J. McKerrow, J. Sakanari, "WormAssay: A Novel Computer Application for Whole-Plate Screening of Macroscopic Parasites", PLoS Neglected Tropical Diseases, Vol. 6(1):e1494, 2012

(2)Computerphile. "Optical Flow - Computerphile." *YouTube*, uploaded by Computerphile, 16 Oct. 2019, www.youtube.com/watch?v=5AUypv5BNbI.

(3)Kitt, Bernd, et al. "Block-Matching Based Optical Flow Estimation with Reduced Search Space Based on Geometric Constraints." *13th International IEEE Conference on Intelligent Transportation Systems*, 2010, doi:10.1109/itsc.2010.5625181.

(4)Raudies, Florian. "Optic Flow." *Scholarpedia*, Scholarpedia, 8(7):30724., 2013, www.scholarpedia.org/article/Optic_flow#A1_Method_of_Lucas_.26_Kanade_.281981.29.