UNIVERSIDADE FEDERAL DE MINAS GERAIS
ELECTRICAL ENGINEERING DEPARTMENT
SYSTEMS ENGINEERING

VIRTUAL DRUM: AN IMAGE PROCESSING DRUM MACHINE

MATHEUS BITARÃES DE NOVAES

Advisor: Prof. Dr. Euler da Cunha Francisco Teixeira
Universidade Federal de Minas Gerais – UFMG

Co-Advisor: Dr. Hani Camille Yehia
Universidade Federal de Minas Gerais – UFMG

BELO HORIZONTE
JUNE 2018

MATHEUS BITARÃES DE NOVAES

VIRTUAL DRUM: AN IMAGE PROCESSING DRUM MACHINE

Course Conclusion Paper presented in Universidade Federal de Minas Gerais as requisit for Bachelor in Systems Engineering.

Advisor — Euler da Cunha Francisco Teixeira
Universidade Federal de Minas Gerais – UFMG

Co-advisor — Hani Camille Yehia
Universidade Federal de Minas Gerais – UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ELECTRICAL ENGINEERING DEPARTMENT
SYSTEMS ENGINEERING
BELO HORIZONTE
JUNE 2018

MATHEUS BITARÃES DE NOVAES

VIRTUAL DRUM: AN IMAGE PROCESSING DRUM MACHINE

Course Conclusion Paper presented in Universidade Federal de Minas Gerais as requisit for Bachelor in Systems Engineering.

Euler da Cunha Francisco Teixeira
Advisor

Hani Camille Yehia
Co-Advisor

Adriano Vilela Barbosa
Guest Professor

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ELECTRICAL ENGINEERING DEPARTMENT
SYSTEMS ENGINEERING
BELO HORIZONTE
JUNE 2018

Resumo

Esta monografia tem como objetivo principal descrever o processo de pesquisa e experimentação do projeto *Virtual Drum*, que é um instrumento musical cujo o funcionamento está baseado em processamento de imagem e de áudio. Tal instrumento tem o objetivo de emular uma bateria eletrônica cujas regiões de contato da bateria poderão ser definidas pelo usuário, não sendo necessário um set de bateria integrado com o sistema.

Palavras-Chave: Visão Computacional; Acompanhamento de Objetos; Processamento de Áudio.

Abstract

This paper describes the process of implementation and experimentation regarding the Virtual Drum, a musical instrument which bases its functioning on image and audio processing. Such instrument has the objective of emulating an electronic drum set in which the user will be able to define the drum regions. Such regions are perceived by the system and by the contact of the drum stick they generate the respective drum sound.

Keywords: Computer Vision; Object Tracking; Audio Processing.

List of Figures

Contents

1 Introduction

Music is, since ancient times, beyond a tool of entertainment, a way of cultural expression. Through music we can extract portraits of societies, identify its aspects and behaviours. Music has the power to portray an image to the outside world and also satisfy emotional needs as outlined by a research done with 2465 adolescents (1149 males; 1266 females; 50 participants did not state their sex) between 13 and 14 years of age who were attending Year 9 at one of 22 secondary schools in the North Staffordshire region of England (NORTH; HARGREAVES; O'NEILL, 2000).

Beyond that, there is strong evidence that musical training can increase brain plasticity, specially in children (HABIB; BESSON, 2009). Also, it was noticed to increase reading and pitch discrimination and speech abilities in 8 year-old children, as demonstrated in (MORENO et al., 2008). They also demonstrate that even short periods of training have significant consequences on the functional organization of the child's brain. That is also stated by (HYDE et al., 2009) which demonstrated that there were structural brain changes after only 15 months of musical training in early childhood. Those changes improved motor and auditory skills.

In Brazil, country of the author, there are not many social projects towards the popularization of musical training. Even so, they demonstrate that such policies have strong effect among the young and poor communities. For example, *Projeto Guri* is a social project which has the objective of recovering, through music, adolescents self-esteem and developing their sense of citizenship.(HIKIJI; NOVAES, 2006)

Such social projects operate based on donations and government incentive. Since the acquisition of a musical instrument and also musical classes are, in the majority of cases, expensive, the popularization of musical knowledge happens mostly through social projects.

This paper has the objective of developing a solution for a low-cost entry level electrical drum set that may offer an option for people that have interest in learning music but do not have monetary conditions to buy an instrument of do not have social programs available near their location.

For that, the paper will be divided into three main chapters: Project, Implementation and Conclusion. *Project* is where a market study is performed and project development tools are chosen and explained, according to what was learned during the Systems Engineering course. *Implementation* is the chapter that will describe all steps and experiments that were performed during this 6 month development and *Conclusion* is the chapter were all results are summed up.

2 Project

2.1 Motivation

It is known that a Course Conclusion Paper is not necessarily motivated by market needs, but it is still interesting to understand the way that people perceive how instrument prices can affect musical learning opportunities and what is the perceived value of an entry level instrument.

So, based on that, a research was performed with 163 Brazilian people (mainly from 20-27) with 6 main questions that were set to understand how they perceive the influence of instrument prices on the musical instrument learning dissemination. Below are the questions and their respective results, translated to English.

- ***Do you play musical instruments?***

Figure 1 – First question

In Figure 1 we see that most people asked (38%) do not know how to play an instrument but would like to learn. 31% already know how to play one musical instrument while 25% know how to play more than one instrument. only 9 people (6%) do not know how to play and are not interested in learning.

So, based on that data, we can perceive that the majority of people asked have interest or contact with musical instruments. So, we can imply that they have the knowledge, experience and interest in answering the next question.

- ***In your perception, what hinders your access to learn a musical instrument?***

In this question, the intent was to understand people's perception regarding what is most difficult for a beginner to get to learn an instrument. More than one option could be chosen in this topic. We can see in figure 2 that 64% allege lack of time, 37% say that it is due to the high price of musical instruments in general and 17%

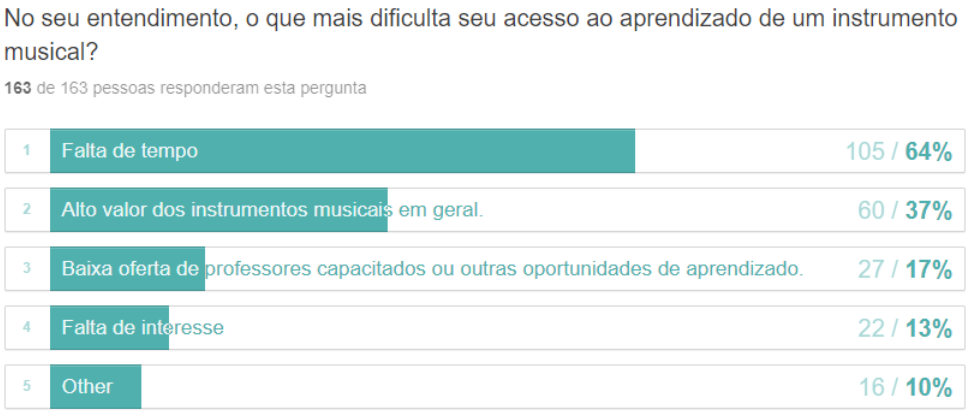

Figure 2 – Second question

state that there are not enough capable teaching professionals. in addition to the 13% that marked "lack of interest", there were also the 10% that marked "other" (and it was an open text), where the majority was "lack of talent, compromise" and high price of musical classes.

We can then imply that the musical instruments price is perceived as an obstacle for learning purposes. This paper proposes the study of a solution for that.

- ***Do you have or play drums?***

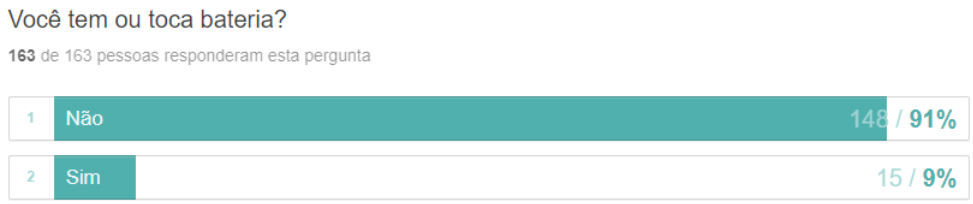

Figure 3 – Third question

By this question we can see (figure 3 that most of people asked (91%) do not play drums.

- ***Do you consider that the drum set price is a limiting factor for the dissemination of this instrument learning?***

  by the result shown on figure 4 it is clear that the big majority (95%) of people asked consider that the price of a drum kit(in Brazil), electrical or not, is a limiter in spreading the knowledge regarding this instrument.

- ***In case you don't have a drum and have interest in learning it, would you pay R$ 200,00 on a electronic drum set?***

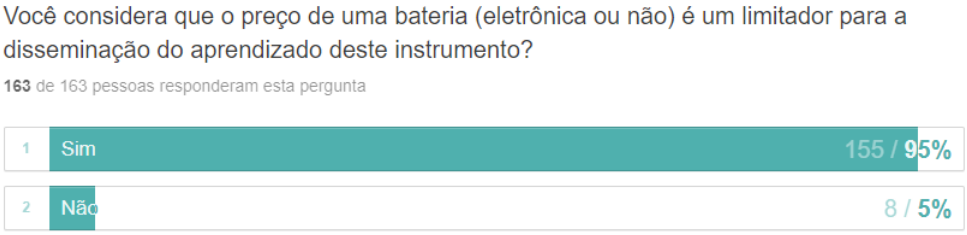

Figure 4 – Fourth question

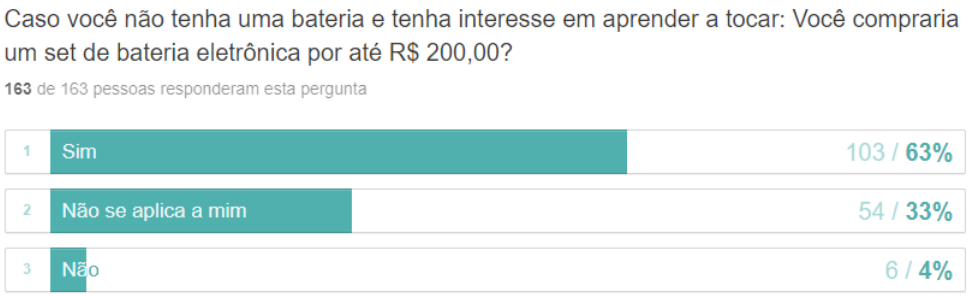

Figure 5 – Fifth question

By the result shown on figure 5, we can see that 63% would be interested on buying a cheaper drum kit for learning purposes, against 4% of negative and 33% of "not applicable to me". This indicates that, if available, a cheaper solution would be greatly appreciated for learning purposes.

- ***Do you consider that the production of an entry level, low-cost instruments with the objective of spreading musical knowledge would be valuable?***

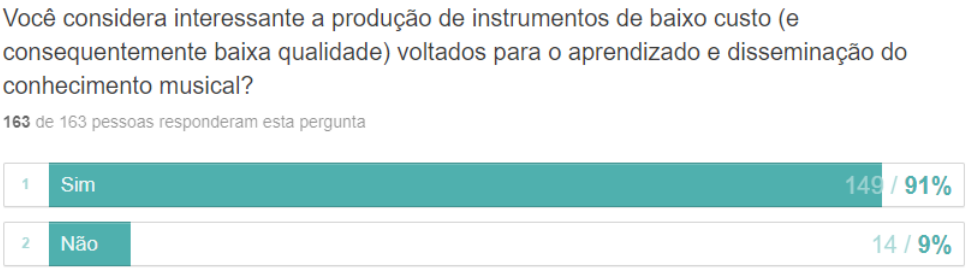

Figure 6 – Sixth question

Through this question, it is possible to notice that most people believe that an entry level instrument is important for spreading the musical learning opportunities, since 91% voted "yes" while 9% voted "no".

An interesting aspect pointed by some that voted "no" is that entry level instruments can interfere in the learning process, making it even more difficult. On the

other hand, the cheaper instruments may be the entrance solution for someone that would never have the opportunity to learn if only expensive instruments were available.

Given the results above, it is reasonable to imply that the market would approve a solution for a cheaper drum set for learning purposes. So, the idea of this paper is to develop a solution that is low-cost, by using current image and sound processing techniques to emulate a drum in a computer or mobile device.

2.2 Proposal

In the previous section, it was shown that a low cost-drum set is considered to be a good solution for learning purposes. So, this paper will describe the development of a software and hardware that fulfill those needs.

The goal is to use image processing techniques and sound processing techniques to detect the drum stick, the regions that would emulate the drum hardware components and the impact of the drum stick in those regions, with its intensity.

2.3 Scope

2.3.1 Project Methodology

The methodology chosen for this project is the spiral model. This model is based on cycles of development in which each loop is an increment of the last iteration. It was created to suit software development/enhancement and its peculiarities (BOEHM, 1988). For this specific project, each round will have a product delivered at its end. And each loop will be an enhancement of the last one, as represented in figure 7

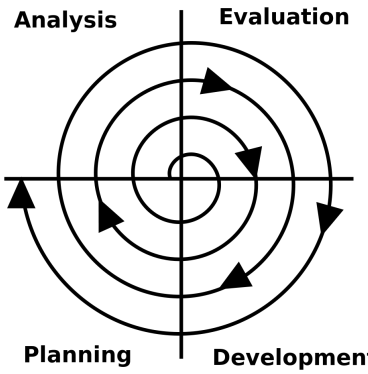

Figure 7 – spiral development methodology

2.3.2 High Level Architecture

The high level system architecture is represented in figure 8.

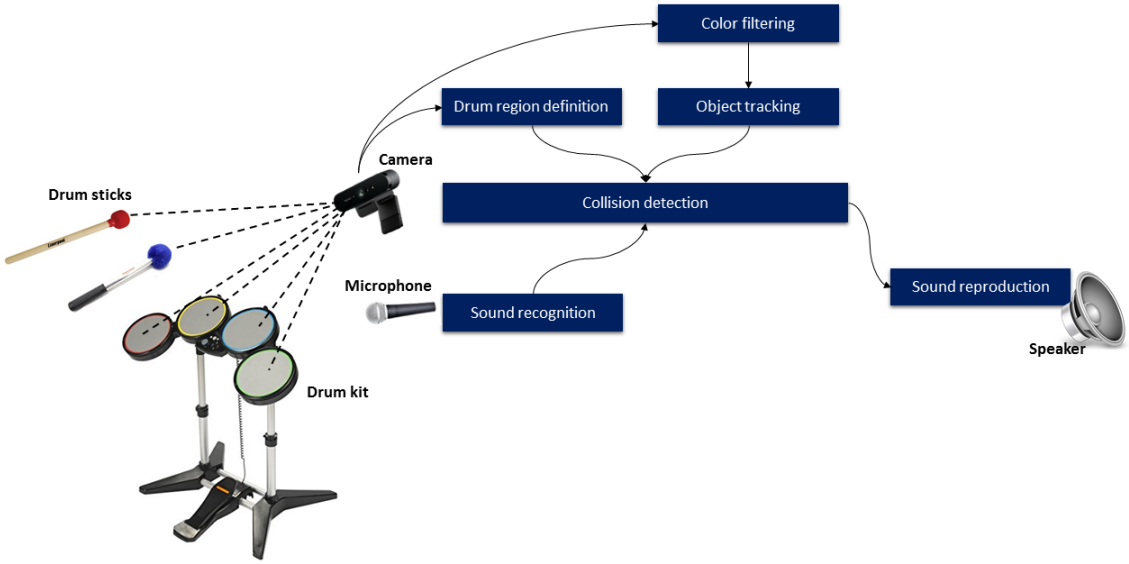

Figure 8 – High Level Architecture

The system uses a camera to get real-time imagery information. The user selects the regions of interest for the drums. Each region represents a distinct drum instrument. The system will then use a color filtering algorithm to detect the drumsticks and track their movement. When there is a sound input with a certain amount of energy (or more), the system checks if the drum stick is inside one of the drum instrument regions. If yes, that instrument's sound is generated.

2.3.3 Cost Estimation

For the purpose of this paper, the system will be developed in a computer. But, thinking as a product, it can also be developed for a mobile phone. both phone and computer costs are not included into the cost estimation. It is assumed that the person that is buying the system already has a computer or a mobile phone. A specific drum set can be developed for enhancing the experience with the application (but it is not mandatory for the correct functioning of the system). Specific drum sticks are needed. So, the costs of the project are for building the drum set and the drum sticks. Doing a market research, R$ 10,00 is a reasonable cost for an entry level drum stick. Regarding the drum set, since it is not the main purpose of this paper, no quoting was made for the inclusion of a plastic drum set. But we can see on (DRUMPADS. . ., 2018) that a kit

with 4 drum pads can be acquired by R$ 129,99. Aggregating this cost information with the response of question 5, we can use a cost estimation of R$ 200,00 for this project.

2.3.4 Timeline

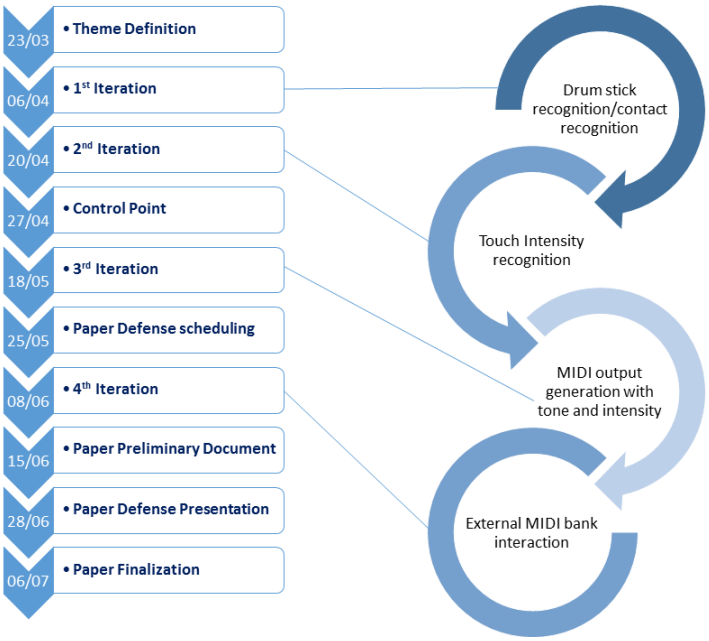

Figure 9 – Initial Timeline

The initial timeline of the development was as shown in figure 9. As the algorithm was being implemented it had to be modified because the latency was not acceptable and it was needed to focus on this matter. The scope of the project also had to be reduced. the second iteration changed to focus on latency improvement and the third iteration changed to focus on the sound processing implementation, as can be seen in figure 10

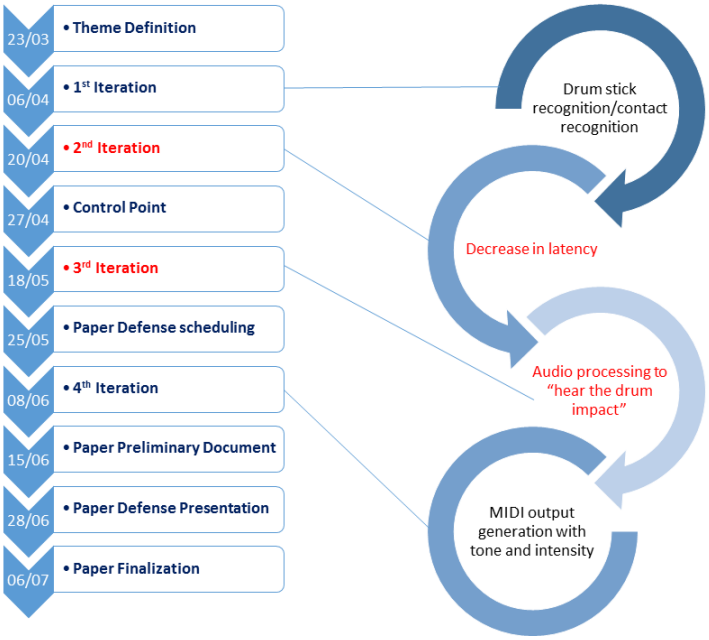

Figure 10 – Updated Timeline

3 Implementation

In this section the progress of the project will be described, such as the main technologies implemented and the evolution of the project development. Three products were generated according to the timeline of figure 10. Each product result will also be described in this chapter

3.1 Initial development definitions

For the implementation, it was decided to use OpenCV (Open Source Computer Vision Library) for the computer vision methods. It is a library that is free for both academic and commercial use. It has interfaces with programming languages such as C++, Python and Java and supports Windows, Linux, Mac OS, iOS and Android. It is widely used for image processing and there are significant content in the internet regarding examples and tutorials. Documentation is available in (OPENCV..., n.d.).

Figure 11 – OpenCV Logo

The programming language chosen is python, which is an open source language, widely used and high level. Also, there are many tutorials for Python + OpenCV implementations in the internet. For an optimized implementation, C++ would be the best language. But python suits well the objectives of this paper. Documentation of python is available in (PYTHON..., n.d.)

Figure 12 – Python Logo

The computer used was a MacBook Pro (Retina, 13-inch, Mid 2014), processor 2,6 GHz Intel Core i5 and memory of 8 GB 1600 MHz DDR3. the camera used was the Macbook integrated webcam, with a frame rate of approximately 30 FPS (Frames per second). The microphone used was the integrated Macbook microphone.

3.2 First Product

The first product focused on having a system which tracks the drum stick tip and recognizes when it enters into predefined video regions (rectangles). When that happens, a *.wav* file is executed, emulating a drum sound. the system architecture and main methods/functions will be explained in the next sections.

3.2.1 System Architecture

The system architecture for the first product is as represented in figure 13. The webcam will be sending imagery data for the "Object tracking" module and the "Drum region definition" module. The "drum region definition" module stores the vertices of the regions defined by the user in the first step (drum region definition) and then displays it in the screen during the second step (drum execution). The "object tracking module" uses color filtering to recognize the blue ball located at the drum stick's tip. After that, the module calculates the location of the center of the blue ball. With those two location information, the "Impact detection" module will then calculate if the center of the blue ball is inside of any rectangle in the screen. If yes, an impact is then recognized and the "Sound reproduction" module will execute the region's *.wav* respective sound

Object tracking, which is made by color filtering and will be explained next.

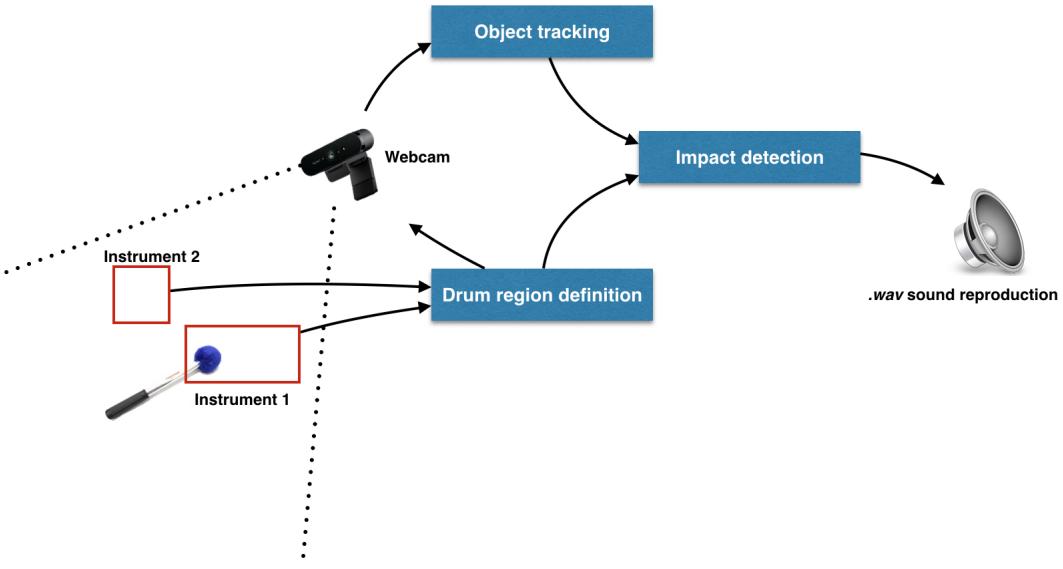

Figure 13 – First Product System Architecture

3.2.2 Drum Region definition

The drum region definition occurs before the drum starts. The video is displayed and the user can select regions (rectangles only) by clicking and dragging in the screen.

After selection, the user can press and hold "S" to confirm the region or just start drawing other region to erase the old one. After selecting all desired regions, the user can then press and hold "T" to begin the drum execution.

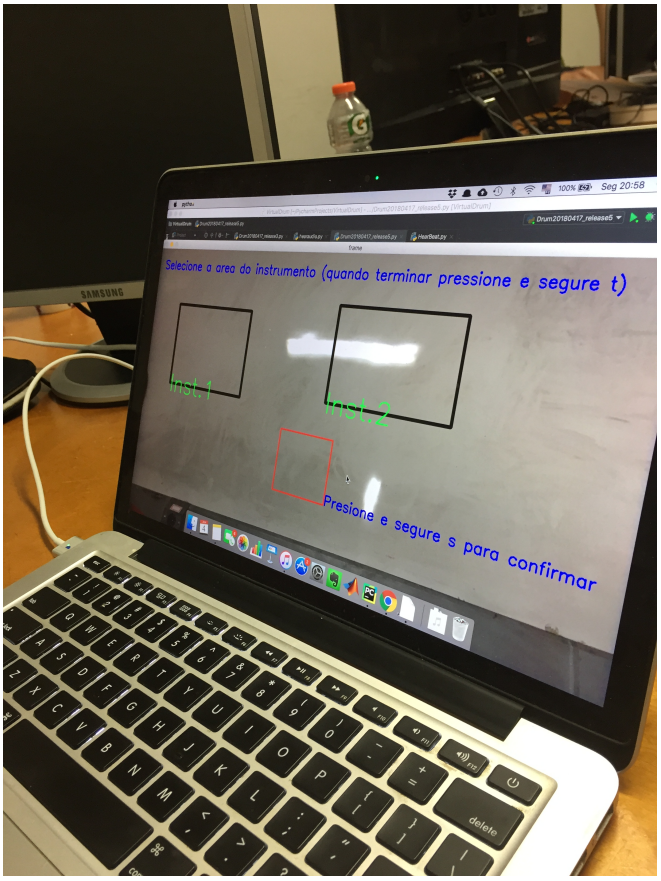

Figure 14 – Drum Region Definition

As can be seen in figure 14, when the region is selected, the rectangles turn from red to black. When a region is not yet selected, a message "Press and hold 's' to confirm" is displayed.

In this implementation there is no instrument sound choice. As the user chooses the regions, there is an automatic sound attribution as below:

- First Region - Snare
- Second Region - Bass Drum
- Third Region - Snare
- Fourth Region - Cymbal

Region attribution is limited to 4 regions, since this initial product is for testing purposes only.

3.2.3 Color filtering (HSV)

To recognize the tip of the drum stick, a color filtering method will be used. Such method will get image data from the camera and turn black all pixels that are not into a pre-defined HSV(hue, saturation, value) range and turn white all pixels that are between this lower and upper boundaries.

HSV is an alternative representation from the RGB color model. It was created in order to provide a more intuitive color scale, based on intuitive appeal of artist's tint, shade and tone. The relation from RGB to HSV happens according to below equations(PLATANIOTIS; VENETSANOPOULOS, 2013):

$$H_1 = cos^{-}1\frac{\frac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}}$$

$$H = H_1, \text{if} B \leq G$$

$$H = 360 - H_1, \text{if} B > G$$

$$S = \frac{max(R,G,B) - min(R,G,B)}{max(R,G,B)}$$

$$V = \frac{max(R,G,B)}{255}$$

The HSV relation can also be seen in figure 15.

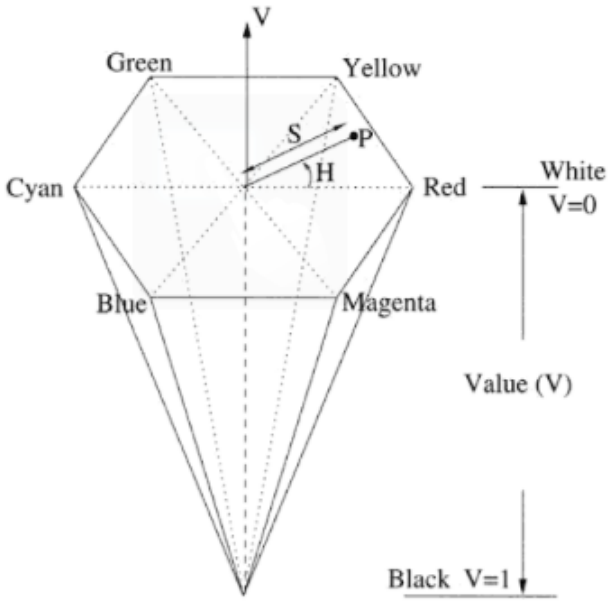

Figure 15 – HSV color space

For the first drum stick tip, the color chosen to be filtered was blue. For the other drum stick (which will be implemented in product 3), the color will be red.

The main reason that the HSV color scale was chosen in this implementation is because there is an Opencv method called *inRange()* which uses such scale. So, once this method is also widely used, there was no need for implementing a new method with RGB scale. Also, doing a deeper research, it can be noted that HSV color works better for color filtering than RGB, once HSV separates the color intensity from the color information, letting such scale more robust to noises (like shadows in the image)(SIGNAL..., 2012).

can be used to make HSV filtering in video frames. The filtered image will be as displayed in figure 16. To enhance filtering quality and remove noise, there are two morphological operations that can be used. Erosion and dilation.

Dilation is about making a convolution of the current mask with a predefined kernel, usually a square or circle. This kernel is then scanned over the image and the maximal pixel value overlapped is stored and applied in the anchor point position. It causes the white contour to be "dilated", as can be seen in figure 16.

Figure 16 – Dilation - taken from (OPENCV..., 2018)

Erosion can be understood as the opposite of dilation. A predefined kernel is convoluted with the current mask and the minimal value overlapped is stored and applied in the anchor point position. This causes the white image to be "eroded", as can be seen in figure 17.

Figure 17 – Erosion - taken from (OPENCV..., 2018)

In the algorithm function *morphologyEx()* will be used. It does both erosion and

dilation in the image. Those transformations combined enhance color filtering quality. As can be seen in figure 18, the original frame contains not only the blue tip of the drum stick but also the blue blades of the ventilator. After the erosion and dilation, those white pixels are erased. The next step is to find the centroid of the blue ball.

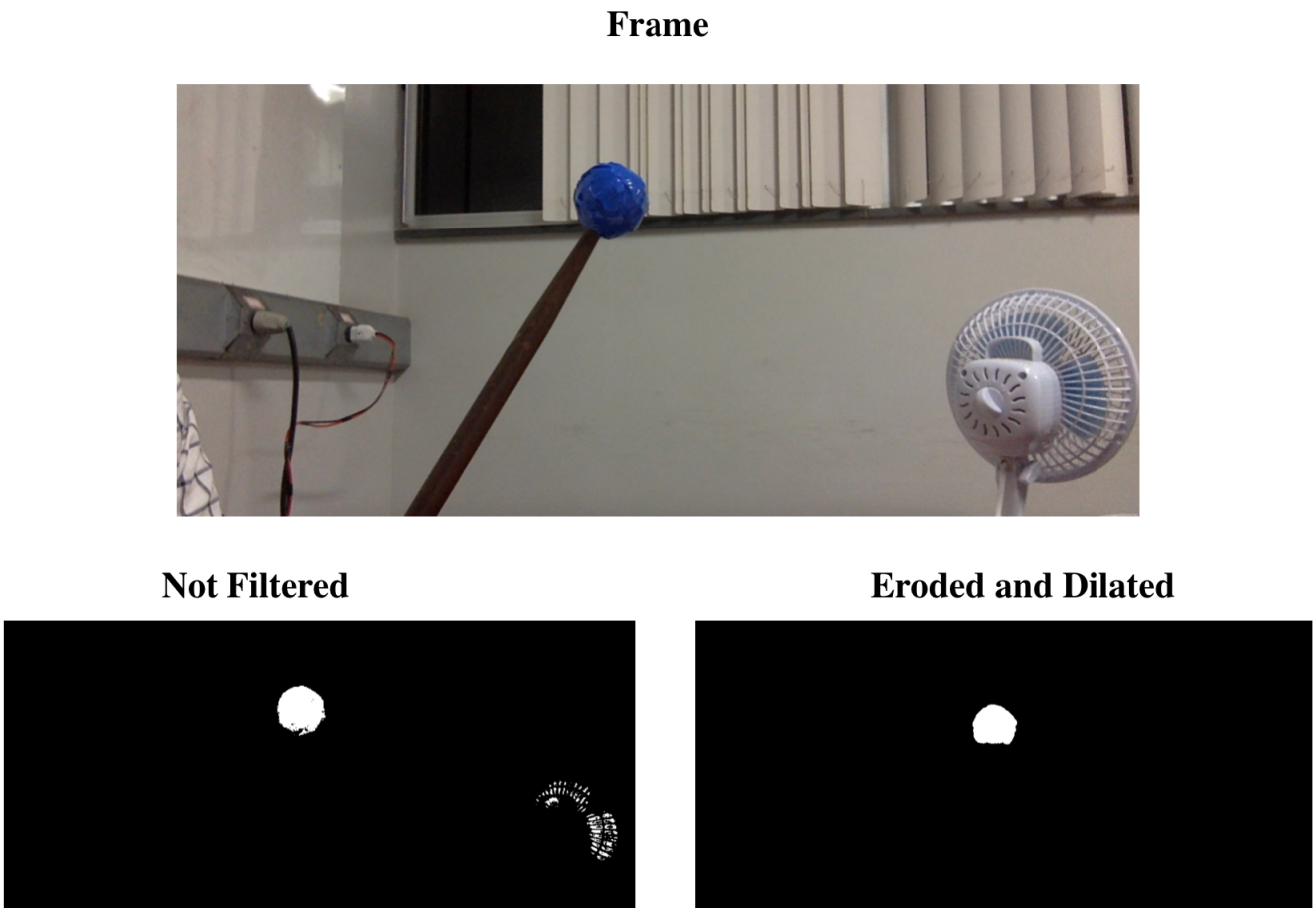

Figure 18 – Comparison between transformed and not-transformed mask

3.2.4 Object tracking

After the mask is filtering the blue tip of the drum stick correctly, a tracking algorithm needs to be implemented. Opencv method *findContours()* is used in order to find all white blobs in the image. This function will then append all white contours found in the filtered mask (that is why it is important to set a good quality color filtering in the previous step). From that, we get the contour with higher area and store its radius. If the radius is higher than a certain value (in this development, the threshold is 10), then a minimum enclosing circle is drawn in the frame, showing that the object was recognized and its centroid is being tracked. This can be seen in figure 19

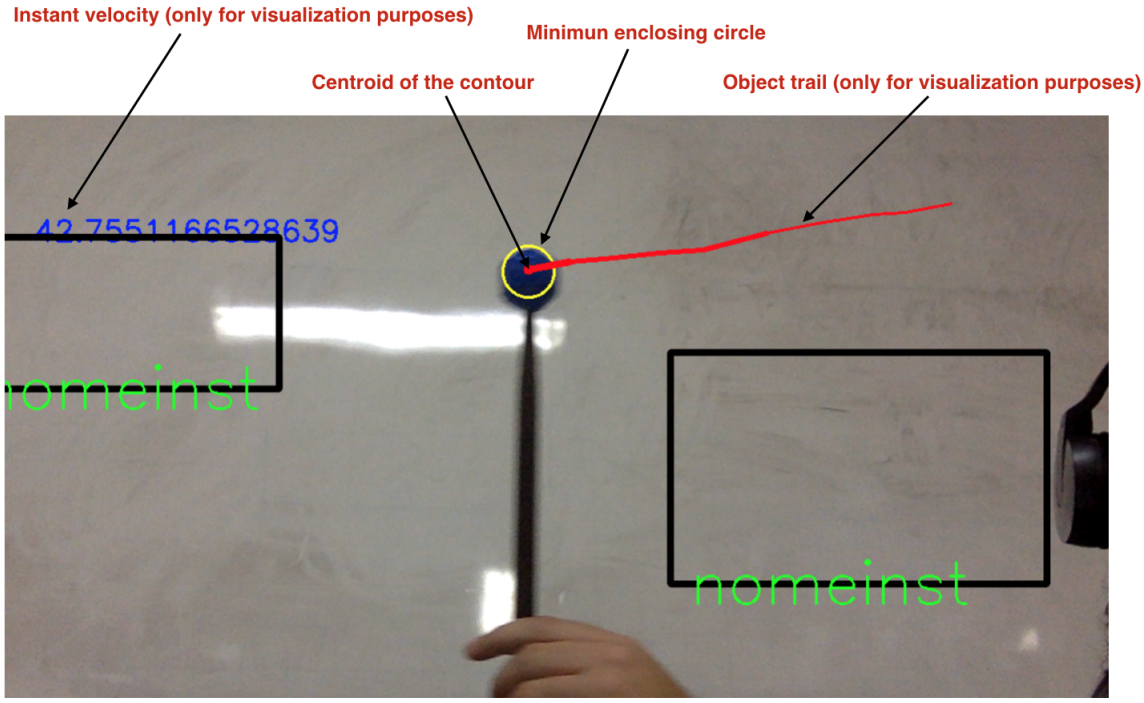

Figure 19 – object tracking

So, with that we have the x and y coordinates of the drum tip centroid for each algorithm iteration. Now, a function *isInsideRegion* was implemented in order to check if the centroid is inside any of the regions defined by the user. If yes, a *.wav* sound is reproduced.

3.2.5 Sound reproduction

The sound reproduction was done by using the library *simpleaudio* in python (available in (SIMPLEAUDIO..., n.d.)). This library allowed the reproduction of simple *.wav* sounds, downloaded from a free sound bank in the internet. So, each time the center of the drum stick is inside any defined rectangle, the sound related to that region is executed once. To execute it again, the drum stick tip has to go out of the region and then go in again.

3.2.6 Results

In this implementation the object tracking worked successfully with a blue tip drum stick. The following goal is to make the system to work with two drum sticks (add another drum stick with a red tip). The region attribution phase was also working correctly. An enhancement could be changing the region attribution from rectangle regions to polygon/ellipsis regions. The sound reproduction also worked as expected. The issue identified was the high latency. During the drum execution,

it takes considerable time for sound execution after the physical contact of the drum stick - region was performed. Such latency is not satisfactory, so the next development iteration is focused on reducing it at its maximum, but keeping the objective of a low cost product.

3.3 Second Product

As described in the results section of the first product, the delay between the sound reproduction and the actual contact between the stick and the drum was not satisfactory. So, in this product, the main goal is to reduce the latency.

3.3.1 Latency Analysis

On real time applications, latency must be decreased to levels that provide an acceptable perceived quality of the product. For drum instruments, for instance, the often accepted latency could not go over 10ms(MÄKI-PATOLA, 2005). Such latency is a challenge to reach with low cost equipment. The 30 FPS camera is a limiter, such as the sound reproduction time. Some strategies will be tested below.

The latency is not mainly affected by the number of instruments in screen or by the object tracking. The experiment below was done executing the drum several times and taking the average time of each step in the loop.

loop average time (ms)	**130.48**
frame read time (ms)	37.65
Contour finding and drawing time (ms)	15.33
Velocity tracking, movement drawing time (ms)	0.20
Check region contact and draw images time (ms)	0.62
Image show time (ms)	6.17
Wait for key time (ms)	70.50
Sound reproduction average time (ms)	19.13

So, we can perceive that the frame read step and the *waitKey()* function are the ones that have the highest time costs. Both are because of the frames read and reproduction. The "wait for key" is a method that is required in python for video execution. It waits 1ms for a key press and also does the actual image show procedure. That is why *waitkey()* function takes more time then *imshow()*(image show function), which created only a mat header, as described in (OPENCV..., n.d.). The actual image generation happens in *waitKey()* step (function that is mandatory for the proper behaviour of Opencv library in Python).

Also, it is possible to check that the sound reproduction time is on average around 19ms, which is not high in comparison with image read/show timing (but still not acceptable for a professional instrument). So, in this second product the main goal is to decrease the image read/show latency. For that, the first strategy was to increase the camera's FPS rate.

The camera that was being used was the MacBook Pro webcam, which has an FPS rate of approximately 30 frames per second. So, one solution could be acquiring a webcam with higher FPS. According to a market search, the following solutions were found:

Webcam	**FPS**	**Price**
Logitech BRIO Ultra HD webcam - USB	90	R$620,00
Razer Stargazer Webcam	60	R$1259,00

The research showed that high FPS USB cameras are expensive and go out of the preliminary budget. And "non-USB" cameras (such as *Go Pro*) with high FPS are also expensive and requires a USB video interface which is also expensive. The *Go Pro* adapter, for example is around R$ 1000,00. So, getting a better quality camera is out of the project scope.

Given that, a possible solution for decreasing the effect of camera's fps rate into general execution latency is to use threads in order to make parallel processing to separate the image read from other operations. Also, since the *imgshow()* method costs significant time, it will be removed. In this way, the algorithm execution latency will be limited only by the camera FPS. After that, in the third product, the audio processing will be used, since audio sampling can be done in much higher sampling frequency.

3.3.2 Threading

Threading is seen as a suitable option for decreasing an algorithm execution time, by increasing the memory used for the execution. For this product, the "video read" procedure will be taken off the main execution and will run into a separated thread, as shown in figure 20.

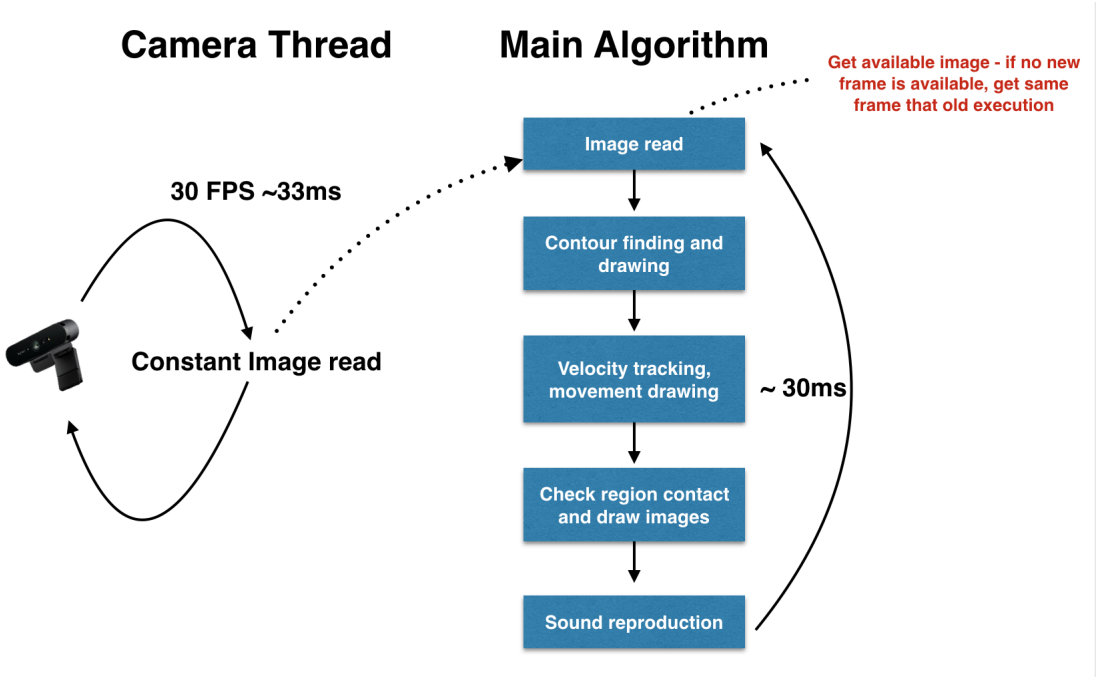

Figure 20 – thread for camera read

The system will then get the current available image and make all calculations with it. Without the thread, the system will be waiting for the new image to be read and then it would continue the execution. For instance, if we use a camera with 1 FPS, the algorithm will take around 1 second for each loop. If we use the thread for the camera read, the algorithm will take 30ms for each loop, which is the summed time for all other operations. But still, around 34 loops will be processing the same image provided by the camera in that one second. So, in a certain way we are still limited to the camera's FPS rate.

3.3.3 System Architecture

Since this second development iteration was mostly about decreasing execution latency, the architecture remained similar to the architecture of figure 14, but with the improvement shown in figure 20.

3.3.4 Results

The threading strategy for the video reading improved the loop time to around 30ms. A reduction on that loop time has to be considered for the next product. But still, once the stick impact is being measured by image analysis (checking if the stick is in the region or out of the region), we are still dependent of the camera FPS rate. Once buying a higher FPS rate camera is out of scope, a change in the algorithm will be required. For the next implementation, audio processing will be added to capture the impact time.

This will allow a faster response and also will allow the output sound volume to be proportional to the input sound volume.

3.4 Third Product

On the third product, audio processing will be added. Also, a second drum stick (with red tip) will be added. The ".wav" sound output will be changed to a sinusoidal signal, which will vary its amplitude according to the intensity of the impact detected. This change is in order to better perceive the loudness variation of the output. The frequency of the output will also vary according to each region defined.

For the proper functioning of this third product, the drum region definition methodology must change. As can be seen in figure 22, the user must now select a region that contains the drum surface and also the possible regions in which the stick will be located before the impact.

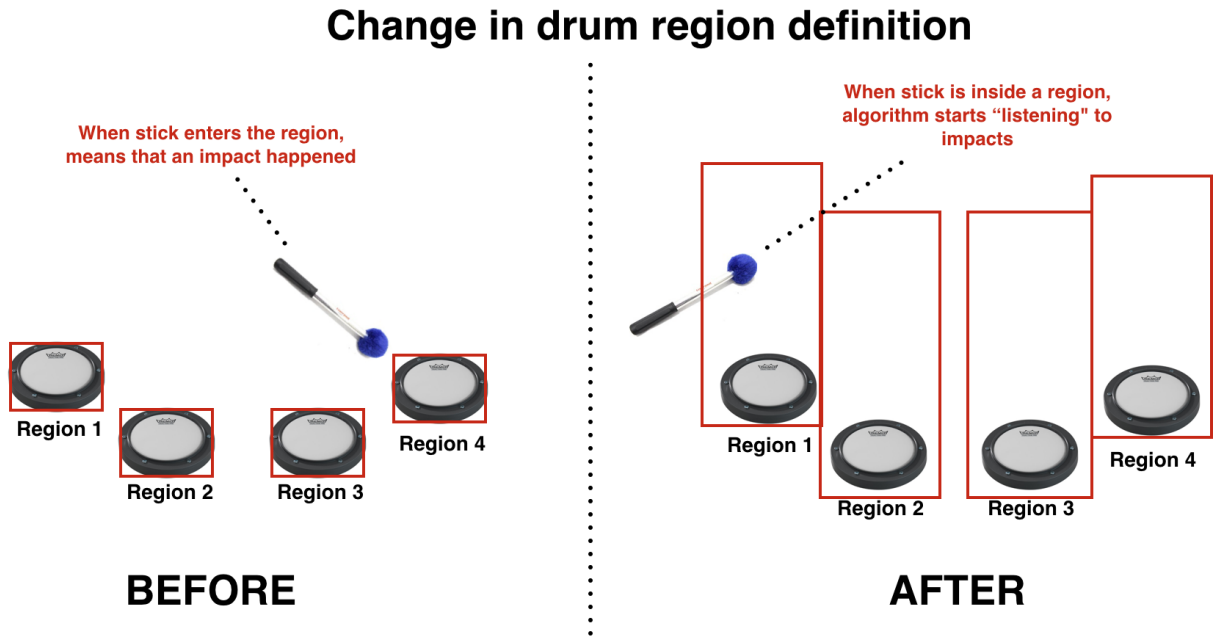

Figure 21 – Difference between drum region definitions. In second/first product, the user should define the region in which the drum surface would be. In the third product, the user must select the region which contains the drum surface and also its surroundings

Now, when the drum stick enters the drum region, the system will start getting audio data in order to check if a collision happened. If there is any impact, the system will generate a sound wave with a volume proportional to the intensity of the shock.

3.4.1 System Architecture

For this third development iteration, the system architecture is similar to the one in figure 13 but with the addition of another thread for sound processing. Such module operates in parallel with the normal loop and the sound analysis is turned on when the main thread identifies that the drum stick is inside one of the drum regions. The representation can be seen in figure 22. The project is now in accordance do the high level architecture shown in figure 8.

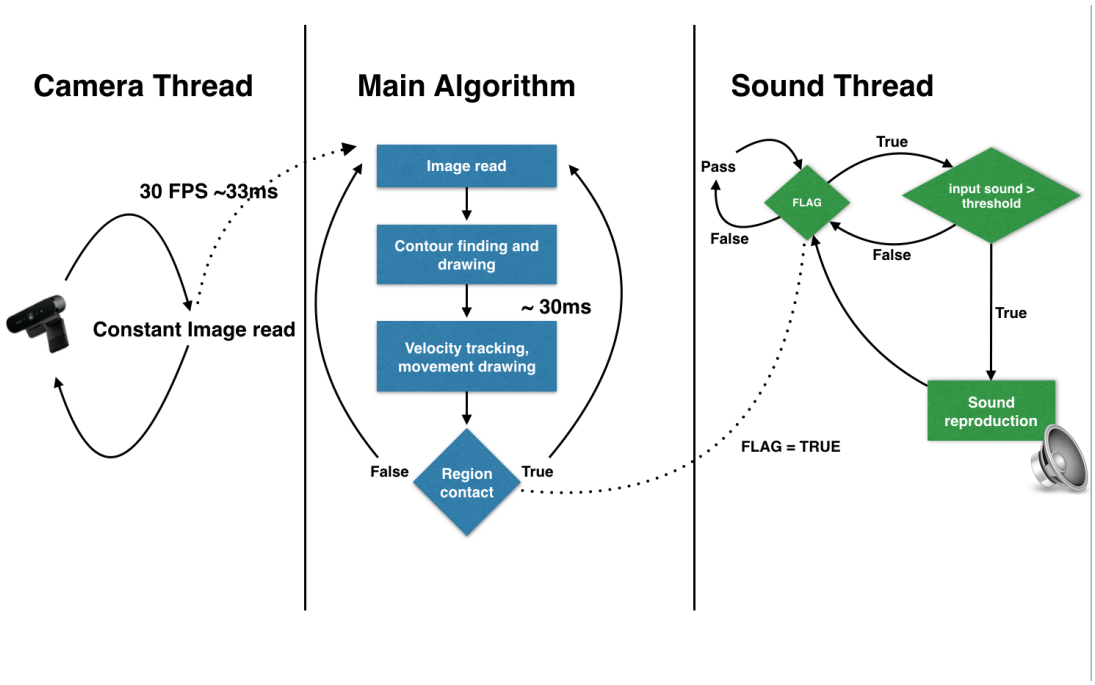

Figure 22 – System Architecture of third product

3.4.2 Sound recognition

For the sound recognition, it was used the library *Pyaudio*, available in (PYAUDIO..., n.d.). This library allows sound recording and playing.

As shown in previous chapter, a separate thread will keep hearing the audio from computer's microphone in order to detect a collision. The collision calculation happens by getting a chunk of audio data (which was defined empirically and may vary from different computers or mobile phones) and calculating its energy. The energy calculation is done by below RMS formula:

$$E_{rms} = \sqrt{\frac{1}{n}\sum_{1}^{n} x_i^2}$$

where,

E_{rms} = RMS energy of the window

$x =$ audio magnitude

$n =$ Size of the audio window

Then, after some tests and tuning, it was decided to use a threshold value of 0.05. So, if the mean RMS value of the sample is above 0.05, the algorithm will play a wave sound to the user and its volume will be as below

$$Vol = \begin{cases} c * E_rms, & \text{if} \quad E_{rms} \leq V, \\ 1, & \text{if} \quad E_rms > V. \end{cases}$$

Where c is a constant that have to be set according to the ambient sound and V is a threshold found experimentally and which varies according the hardware and microphone used.

3.4.3 Results

The sound reproduction latency had decreased significantly once the sound processing and energy calculation is faster than the image processing with a 30 FPS camera. And putting the sound processing in a different thread avoids that the sound check gets limited by any other logical operation in the main loop. Curiously, the latency in the main loop has increased considerably, reaching peaks of 800ms.

Also, this new way of defining the drum regions can cause some limitations in the project execution. For example, if the drummer suddenly changes from one surface to another and hit the instrument, the sound that will be executed may be the one from the previous instrument, as the region recognition is still dependent on image processing.

3.5 General Results

This development demonstrated that it is feasible to build such an application. It still need improvements to become a commercial product but, since this is not the main goal of this Conclusion Course Paper, it can be considered that the development was successful and reached all the objectives stipulated during the second timeline (timeline that was revised after the latency issue was noticed).

It is important to mention that the drum stick tracking was working in a very satisfactory way since the first implementation. Also, the region definition could be implemented successfully since the initial product. The initial development iteration was concluded with good results. That was also when the latency issue was identified. So, the next product was mainly about using a thread for video frames acquisition

in order to decrease latency. It was then perceived that the main latency bottleneck for the system was the camera FPS. So, as a higher FPS camera costs more than the project budget scope, a new solution had to be developed, which was using sound processing for "hearing" the collision of the drum stick with the region. With that, the sound reproduction got faster, almost approximating a suitable commercial product. The curious point is that, even using threads, the average main loop time raised considerably (sometimes 800ms per loop). This is probably due to thread synchronization.

We can consider that the development achieved the objective proposed in this paper. But, as stated before, it needs improvements to become suitable for commercial/personal use. Such improvements will be described in the next section.

3.6 Next steps

After the third product, there are still enhancements that can be performed in the system. One of them is developing a mobile phone app that will use the integrated camera and microphone to operate. This development can happen in a lower-level programming language, enhancing then the processing optimization and helping to decrease latency. Also, some mobile cameras can operate in a FPS rate higher then 30 (FPS used in this development), which also improves the latency. Apple's IPhone 6, for instance, has a camera that can reach 120FPS.

Another improvement is to generate MIDI output instead of a sinusoidal signal with a certain frequency (depending on the region) and a certain volume (depending on the impact). Together with the MIDI generation signal, it could be integrated with software such as Apple's DAW (digital audio workstation) GarageBand. This would allow the selection of a range of good quality drum sounds that will be able to be played through the application.

Another step is to quote and build a low-cost physical structure to emulate the drum instruments. The system should then be optimized to work with such a structure, with regions predefined. This would then become a complete drum set solution with low cost for entry level music students.

Another possible enhancement, after the drum set is constructed and a mobile app is created, is to use the own drum set vibration do check for drum stick collisions. The mobile phone's accelerometer can be used to detect vibrations and cross check the collision with the sound and video inputs.

4 Conclusion

Through this Paper, it was possible to better understand the operation of the Virtual Drum project, including its development procedures, phases and methodologies. By the end of this study, the author was able to explore computer vision techniques, sound processing techniques, python programming, project methodologies, parallel computing and market research strategies.

As a product, Virtual Drum still requires enhancements, mainly regarding latency. But as a proof of concept, Virtual Drum demonstrates that, with proper engines, it is possible to make a drum that does not require any physical structure other than the drum sticks. For learning purposes, such a project can provide a low cost solution in order to disseminate musical knowledge and instigate people that would not have the opportunity to learn using an expensive instrument.

Bibliography

BOEHM, Barry W. A spiral model of software development and enhancement. **Computer**, IEEE, vol. 21, no. 5, pp. 61–72, 1988.

DRUMPADS. [**sinelocosinenomine**]. Address: <https://produto.mercadolivre.com.br/MLB-694461297-kit-com-4-pads-de-estudo-bateria-10-polegadas-frete-gratis-_JM>. Visited on: Aug. 6, 2018.

HABIB, Michel; BESSON, Mireille. What do music training and musical experience teach us about brain plasticity? **Music Perception: An Interdisciplinary Journal**, University of California Press Journals, vol. 26, no. 3, pp. 279–285, 2009.

HIKIJI, Rose Satiko Gitirana; NOVAES, Sylvia Caiuby. **A música e o risco: etnografia da performance de crianças e jovens participantes de um projeto social de ensino musical**. [**sineloco**]: EdUSP, 2006.

HYDE, Krista L et al. The effects of musical training on structural brain development. **Annals of the New York Academy of Sciences**, Wiley Online Library, vol. 1169, no. 1, pp. 182–186, 2009.

MÄKI-PATOLA, Teemu. Musical effects of latency. **Suomen Musiikintutkijoiden**, vol. 9, pp. 82–85, 2005.

MORENO, Sylvain et al. Musical training influences linguistic abilities in 8-year-old children: more evidence for brain plasticity. **Cerebral Cortex**, Oxford University Press, vol. 19, no. 3, pp. 712–723, 2008.

NORTH, Adrian C; HARGREAVES, David J; O'NEILL, Susan A. The importance of music to adolescents. **British Journal of Educational Psychology**, Wiley Online Library, vol. 70, no. 2, pp. 255–272, 2000.

OPENCV Forum. [**sinelocosinenomine**]. Address: <http://answers.opencv.org/question/52774/waitkey1-timing-issues-causing-frame-rate-slow-down-fix/>.

OPENCV Tutorials. [**sinelocosinenomine**], Nov. 6, 2018. Address: <https://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html>. Visited on: Nov. 6, 2018.

OPENCV Website. [**sinelocosinenomine**]. Address: <https://opencv.org/>.

PLATANIOTIS, Konstantinos N; VENETSANOPOULOS, Anastasios N. **Color image processing and applications**. [**sineloco**]: Springer Science & Business Media, 2013.

PYAUDIO Library. [**sinelocosinenomine**]. Address: <https://people.csail.mit.edu/hubert/pyaudio/>.

PYTHON Website. [**sinelocosinenomine**]. Address: <https://www.python.org/>.

SIGNAL Processing. [**sinelocosinenomine**], June 22, 2012. Address: <https://dsp.stackexchange.com/questions/2687/why-do-we-use-the-hsv-colour-space-so-often-in-vision-and-image-processing>.

SIMPLEAUDIO. [**sinelocosinenomine**]. Address: <https://simpleaudio.readthedocs.io/en/latest/>.