Networked Software Systems Laboratory

Technion

Project Report

Application Assisted Popcorn Preparation

Students

Or Zamir 313602831

Igor Berendorf 304475791

Instructor

Roee Mazor

Spring 2017

Table of Contents

Figures

Tables

Equations

1 Project Description

Smartphones are helping us perform a variety of everyday tasks. One task that has eluded the capabilities of a smartphone is the art of making Popcorn. In this project, we investigated different methods to analyze paper-bag popcorns spinning in a microwave to conclude when it is ready. We suggested, investigated and compared different methods for completing this mission using a simple smartphone.

The final product of this project is an app which performs the above, while presenting the different methods and their results.

This Project Combines:

- Preliminary Research
- Android Application Development
- Signal Processing
- UX

2 Work Environment

2.1 Hardware

This project's environment in terms of hardware is quite straight forward – we used two Android based phones (Lollipop 5.0 and Lollipop 5.1) to implement and test the application. The phone's microphone is used to record the sound of the popcorn making process. Several microwaves were used for testing, with different power settings, to adapt the application to different power settings (reflected in cooking time duration).

2.2 Software

Preliminary research and graph plotting was done in MATLAB. The application was developed in Android Studio to support devices with Android Lollipop 5.0 and up.

3 General characteristics

The first thing required to understand before approaching this challenge is to understand the characteristics of a popcorn bag and a popcorn kernels. Using our data set (which will be discussed later) we investigated those characteristics. the data collection process was done mainly manual. The relevant characteristics found are the following:

- Each popcorn bag contains about 510 kernels with deviation of 15 kernels.
- When making popcorn, it is expected that about 40 kernels will stay un-popped. Further cooking will result in burning popped kernels and bad taste.
- Each kernel pop duration is about 30 milliseconds. This information is very important in pop detection methods.
- Pops distribution over time is consistent and behaves the same – quiet at the beginning (very few pops) and a Gaussian form when major popping action is taking place. The popcorn is done at the Gaussian's decay. The following figure is an average distribution found by investigating many recordings:

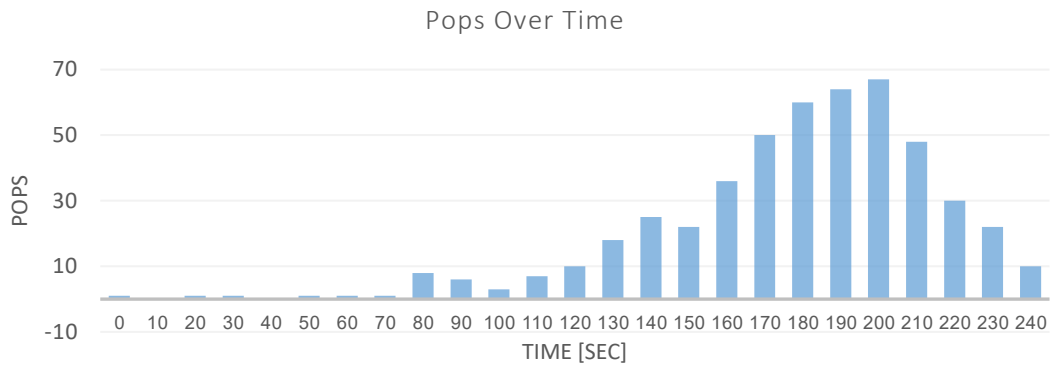


Figure 1 - Pops distibuiton over time

4 Data Set

To perform proper research, we built a data set of recordings. The data set was built in 3 phases and was designated for different uses:

- **10 full recordings of popcorn preparation**, 2-4 min long, depending on microwave cooking power. Those recordings used for the next phases and for real-time testing with the application.
- **Manual marking of popcorn pops** – by visual or hearing. Was used to compare detection methods results (detections) with real pops. There are about 2000 pops marked in the data set.
- **Pops distributions based of marked pops** – were used to compare detection methods results with real pops. Help us have a better understanding of the bag's behavior during preparation.

5 Detection Methods

5.1 Peak\Pulse Detection in Time Domain

5.1.1 Moving Average (Low-Pass)

This method calculates the ongoing average up to some time-point t. Later this average is multiplied by an experimental factor and the outcome is the final threshold, adapted for any time-point. A sample is considered a pop if its amplitude is higher than the calculated threshold of this time-point. The method relays on the fact that most of the time there is only noise in the signal, and this noise is relatively low-amplitude.

Moving Average threshold formula:

$$average(n) = \frac{average(n-1) \times (n-1) + sample(n)}{n}$$
$$Thresh(n) = factor \times average(n)$$

Equation 1 - Threshold calculation in Moving Average

5.1.2 Z-Score

An improvement on Moving Average. For each time frame calculate the mean and variance of the samples. Each sample that exceeds the mean by N*sigma (N is natural, and a selected parameter), is marked as a pop.

5.1.3 Zamir-Berendorf Threshold Derivative Detection (High-Pass)

Based on Threshold Derivative Detection under the assumption that the first samples are noise. This method calculates the white noise average in the beginning of the recording, then subtracts it from all new samples (max(0, newVal)). Later, calculates second derivative in order to sharpen the pops and remove more noise. Finally, an experimental threshold is applied and any sample with amplitude higher than the threshold is marked as a pop. The rationale behind this method is that pops are relatively fast signals, so derivating the signal will emphasize the pops from its background slow background noise.

5.1.4 Energy Peak Detection

This method is widely used for sound analysis. First the average energy of the whole window is calculated. Then, for every sample, the instant energy is calculated (for a window much smaller). Is the instant energy at sample n is higher than the average energy, sample n is marked as a pop.

$$E_{signal} = \frac{1}{N}\sum_{n=1}^{N}\left|x[n]\right|^2$$

Equation 2 - Energy calculation of a window with length N

5.2 Pattern/Interval Detection

5.2.1 Convolution Based Pattern Matching

By finding an average pattern on previous recordings, define the Pattern-to-Match. After every sample window, run a convolution with our Pattern-to-Match, and designate windows as pops if they satisfy some threshold.

5.2.2 Periodicity Detection Using Autocorrelation

The autocorrelation sequence of a periodic signal has the same cyclic characteristics as the signal itself. Thus autocorrelation can verify the presence of cycles and determine their duration. By using this method we can detect recurring sounds in the sample and determine whether a pattern fits an “almost-ready” pattern of popcorn preparation.

5.3 Frequency Domain Analysis

5.3.1 FFT Based Frequency Search

Similarly to Periodicity Detection Using Autocorrelation, we try to identify a frequency pattern that matches a known pattern of “almost-ready” popcorn popping by applying FFT transform on the signal.

5.3.2 Periodogram Based Frequency Search

This is another frequency domain method that is more accurate for low frequencies. This method applies FFT on the autocorrelation of the signal. The autocorrelation extracts the time-domain pattern of the signal, and then this pattern is translated into the frequency domain using FFT.

6 The Algorithms

6.1 Pop Counting Based (Time Domain)

This first algorithm we chose to test is based on pops counting. This algorithm assumes that there is an optimal number of popcorn kernels pops for which the bag is done. To be precise,

the algorithm raises a flag for success when an optimal percentage of popcorn kernels popped in the bag, out of all the units in the bag.

6.1.1 Defining Success Criteria

In order to define the success criteria, the optimal percentage has to be known. After several experiments with popcorn bags, we found that a bag has 510 popcorn kernels in average, with a deviation of 7 kernels. A successful bag (maximum kernels popped and not overcooked), as tested, has an average of 470 popped kernels, with a deviation of 10 kernels, meaning that the "golden zone" is 460 to 480 popped kernels.

6.1.2 The Algorithm

The algorithm is very simple and easy to understand. A chosen method receives an audio recording and extracts the number of pops. The inputs of the algorithm are different for each method, and it is implementation dependent, but the common inputs are:

- Window_Width – defining the sampling window size (by samples or time units).
- Pop_Width – defining the average/nominal width of a single pop in samples or time units. This input's main goal is to avoid double detections of the same pop.
- Threshold_Factor – defining a threshold multiplication factor for detection. For example, is the method outputs an average of the samples, a pop will be detected for signal that its amplitude is higher that Threshold_Factor*average.

Some of the methods require more inputs, as will be broadly described later.

6.2 Interval Based (Time Domain)

This algorithm is based on the old school way for determining if the popcorn is done, as recommended by the manufacturers themselves – waiting until there are 2-3 seconds between pops. We extended this approach to a more general observation – seeking for a time-domain pattern of a well-cooked popcorn bag. Methods for this approach will be discussed later.

6.2.1 Defining Success Criteria

As mentioned already, the Success Criteria in this case is as recommended by the manufacturers – waiting until 2-3 seconds passes between pops. This is the simplified criteria, which we expended to a broader criteria of matching a specific time periodicity pattern.

6.2.2 The Algorithm

We checked two different methods for evaluating if the popcorn is done according to this approach. The first is to measure the time between 2 detected pops, using a pop-detection method. When the interval is sufficiently high, the algorithm declares the bag as ready. The second is to identify a time pattern by finding the periodicity pattern of a sampled window. If the pattern matches is classified as a good pattern, the algorithm declares the bag a ready.

6.3 Frequency Analysis Based

The last algorithm tested is also pattern-based – but in the frequency domain. Assuming there is a pattern in the frequency domain that matches the time-frame in which the popcorn is ready, we can find it. This is similar to the previous algorithm, but requires careful analysis and processing of the signal.

The first approach is to identify pattern in high frequencies (relative to the pops frequency) by finding packs of frequencies that is related to the pops. Each pop is about 20ms long, so we will expect to find noticeable changed in the frequencies about 500Hz. The second approach is to identify the pattern in low frequencies – those that are close to the pops' frequencies.

6.3.1 Defining Success Criteria

The success criteria in this case is pattern-matching base, such as detection of a specific wanted frequency or a sec of frequencies. The actual pattern that classified as a reference for a match is determined after analyzing known recorded signals. A comparison between a "bag is ready" time-frame to a "not yet ready" and "overcooked" time-frames.

6.3.2 The Algorithm

We tested two methods for evaluating if the popcorn is done according to this approach. The first is by applying FFT transform on the signal in order to detect frequency patterns that will indicate the bag is ready. The second method is Periodogram, which is used in the same way. We hoped to detect a change in the frequency pattern when the bag is done or close to be done.

7 Preliminary Results using MATLAB

7.1 Introduction

In this phase of the project we tested all algorithms and methods in MATLAB environment. This phase designed to be eliminate inferior methods and algorithms and calculations-hungry ones. The results are showed for each method as presented in this chapter

7.2 Definitions

The test results are presented by:

- Precision – a measure of how much of the detected pops are relevant. Defined as following:

$$Precision = \frac{No.of\ true\ positive}{No.of\ true\ positive + No.of\ false\ positive}$$

Equation 3 - Precision definition

- Recall – a measure of how much of the real pops are detected. Defined as following:

$$Recall = \frac{No.of\ true\ positive}{No.of\ true\ positive + No.of\ false\ negative}$$

Equation 4 - Recall definition

7.2.1 Frequency Based

In this method we tried to extract a noticeable pattern in the frequency domain that will indicate the bag is done. The results showed us it is hard – up to impossible – to find such pattern.

For the first approach, high frequencies, as showed in the figure below, there is no noticeable change for different parts of the recording – 3 minutes and 50 seconds is about the time the popcorn is done. In addition, many parasitic frequencies are added to the signal, possibly from the phone's electronics, communications, folded frequencies from

higher domain and so on. Those parasitic frequencies are most noticeable on 150Hz, 250Hz and around 450Hz, and they may hide valuable information.

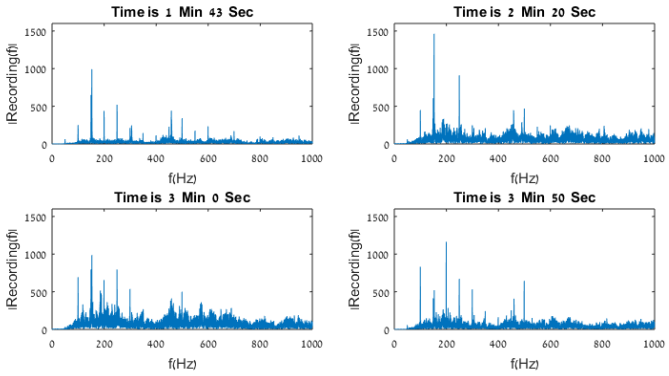


Figure 2 - Frequency domain analysis - high frequencies

As for the second approach, in both methods FFT and Periodogram, the data in those frequencies is very random and without any pattern, so no valuable information can be extracted.

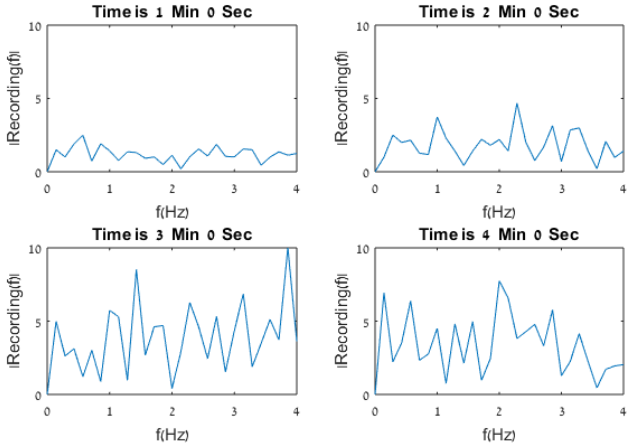


Figure 3 - Frequency domain analysis - low frequencies

To conclude, this method of identifying a pattern in the frequency domain hasn't showed any benefits and was abandoned in this point.

7.2.2 Pattern/Interval Based

This algorithm was also tested using 2 approaches. The first one is by using Periodicity and extracting the interval between 2 pops. This method introduced good results and successfully identified the interval. The figure below presents the product of the Periodicity method for several time points through the recording:

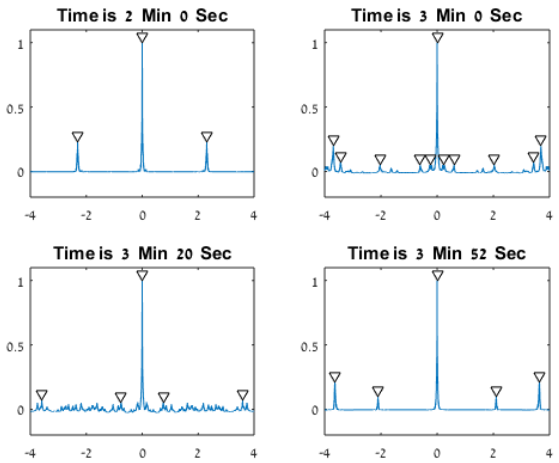


Figure 4 - Periodicity algorithm

Although this method successfully detects the desired interval, it also sensitive to noise and may generate false positive. In that case the algorithm will fail. In addition, this method includes convolution, a calculation-hangry operation that may overload the application or take a lot of time to perform, and therefore we would rather use a simpler method for interval detection.

The second approach is to calculate the interval by measuring the distance between two adjacent detected pops. This approach uses a pops-detection algorithms as mentioned earlier in this paper, such as Moving Average, Zamir-Berendorf, etc. This method proved to be working very well, since we were able to achieve good pop-detection results using those methods. The numeric tests results are presented in "Pop Counting Based" part of the results. We implemented this approach in the application and got good results as well, as will be presented in the "Application Results" chapter.

7.2.3 Pop Counting Based

This algorithm was tested with several methods. The results are in terms of Precision and Recall. The Precision parameter indicates how precise are the results, meaning how much of our detections are relevant (are real pops). The Recall parameter indicates how much of the real pops are detected (or how many were missed). MATLAB results are presented in the following table:

Method	Recall	Precision
Zamir-Berendorf (Derivative)	**85.3%**	**98.0%**
Moving Average	**79.9%**	**97.2%**
Convolution	76.8%	93.6%
Z-Score	71.8%	94.3%
Energy Peak Detection	67.2%	92.1%

Table 1 - MATLAB research results

As can be observed, Zamir-Berendorf and Moving Average methods provided the best results in terms of Recall and Precision. The worst method tested was Energy Peak

Detection – both in parameters and in calculations' complexity. The best two methods are also calculation light and doesn't have additional memory requirements (in addition to the recording itself). Therefore we chose to implement and test both methods in Java/Android environment.

7.3 Pops Detection Example

The process of pop detection in most methods is similar:

- Applying absolute value on the signal – the pop signal is symmetrical around x axis so half of the data is redundant or even bad for the algorithm (for example when calculating average).
- Noise cleaning – useful in most methods to improve results and distinguishing between pops and noise
- Calculations on signal – the "heart" of each algorithm, its goal is to emphasize the pops or adjusting the threshold to the changing signal (with noise, for example)
- Applying threshold – anything above threshold is marked as a pop. At this point, in order to avoid multi detections of the same pop, a parameter called "pop_width" is provided. If a pop has been detected, no other detection will be marked in a distance less than "pop_width",

Figure 5 - Pop detection typical flow

An example of the flow for Moving Average algorithm with the original recording as showed in the figure below:

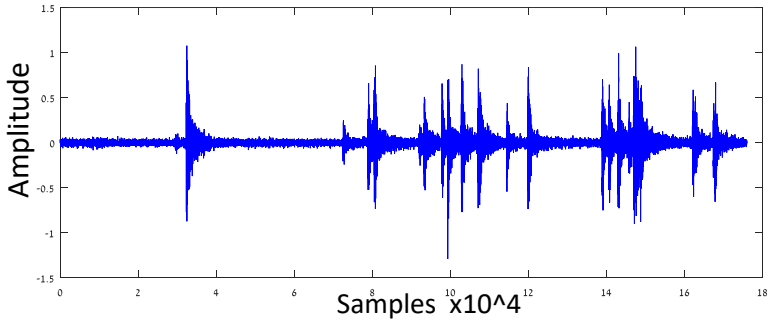


Figure 6 - Moving Average example - original recording

An absolute value is applied on this signal, then the threshold is calculated based on the moving average algorithm. This threshold is the calculated average multiplied by a constant factor, determined by experiments. After applying the threshold, the algorithm mark the relevant detections.

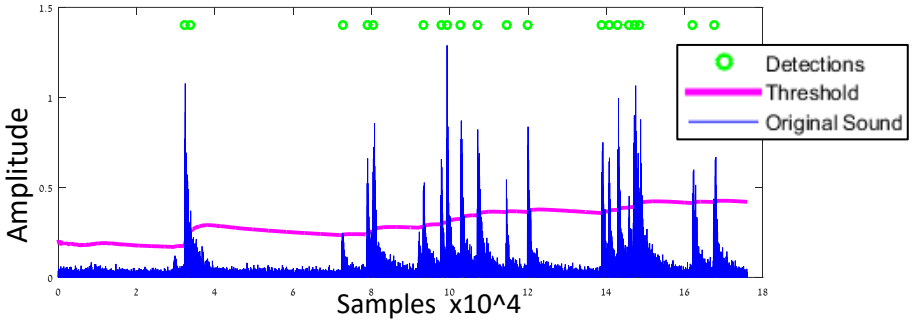


Figure 7 - Moving Average example - detections

7.4 Noisy Environment

One major issue we encountered during research was noise removal. The preliminary research of all methods was under conditions of an almost white noise, generated by low background and microwave noise. After eliminating most of the methods and extracting the best two, we tested them in noisy environment. The noise includes talking near the microwave, moving chairs, closing doors, i.e. regular noises that is expected to interfere the recording in reasonable kitchen environment. Performance were indeed degraded a bit but most of the detections remained the same.

Example of noisy signal with Moving Average detection – as can be observed, a talking noise at the beginning of the window is avoided by increasing the threshold (calculated by the method). Pops are still detected because they are about the threashold.

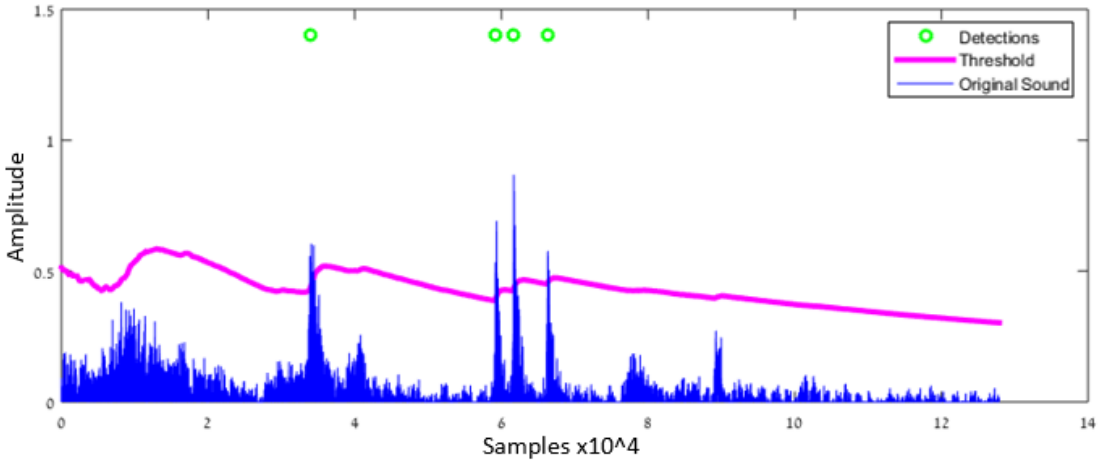


Figure 8 - Moving Average in noisy environment

Example of noisy signal with Zamir-Berendorf detection – as can be observed, at the beginning of the window the noisy signal is originally (red) higher than the threshold line, but after applying calculations (blue) noise is reduced and does not cross the threshold line any more. Pops are still detected as expected.

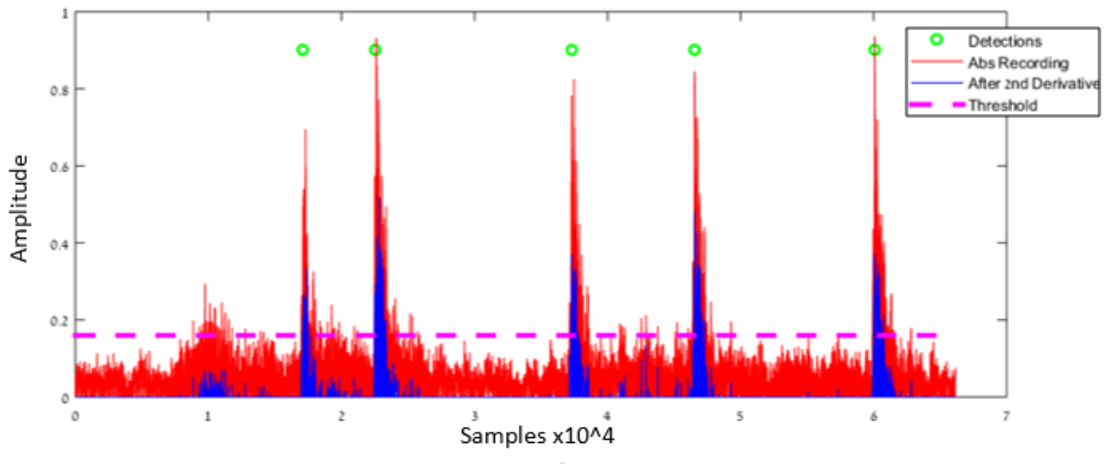


Figure 9 - Zamir-Berendorf in noisy environment

8 Full Bag Detection

8.1 General

We investigated the process required to detect when a bag is done. At first, the approach was counting pops and notifying user when a desired pop count is reached. Quick we discovered this approach is very sensitive to noise and to overlaps between pops, so we completely changed the approach and divided the process to 3 stages. Our main considerations are sensitivity (less sensitive) to errors, saving processing power and accuracy. In this chapter we will discuss the selected process.

8.2 Detection Stages

The detection flow is divided to 3 stages:

- Idle
- Looking for peak (Gaussian peak)
- Looking for interval.

Those stages are required for good and accurate detection with the minimum processing usage and errors. The flow is presented in the following diagram:

Figure 10 - Detection flow diagram

The flow is parameter dependent – the duration of the stage or the algorithm used to mark pops. For a better understanding one can observe the following figure, which connect the flow with the distribution of pops over time:

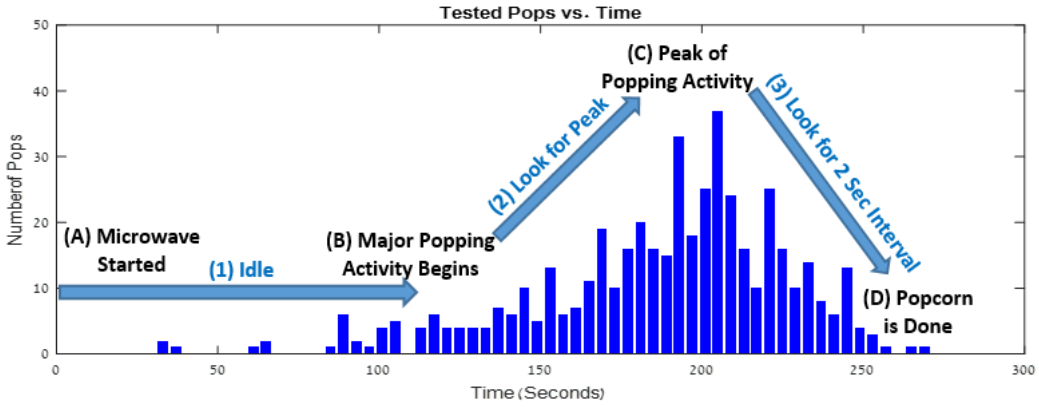


Figure 11 - Bag detection stages

8.2.1 Idle

This stage starts when the microwave starts and the user click "Start" button. The assumption on this stage is that there is almost no pop activity at the time and therefore no calculations are required. This stage "waits" until the Gaussian starts to grow and major pop activity begins.

The duration of this stage is determined by the user's input for microwave power – for low power setting the duration is 2.5 minutes and for high power setting the duration is 1.5 minutes. After the stage is done, the application continues to the next stage – "Looking for Peak".

8.2.2 Looking for Peak

This stage starts right after idle stage. The assumption is that the bag is done always after the Gaussian peak is reached, therefore looking for 2-seconds interval before the peak may introduce errors and false detections. In this stage the algorithm counts pops in 4-seconds windows, and compares the current window count to the last maximum found. If the current count is higher than the last maximum, the maximum value is updated and the search continues. If after 5 iterations (5 windows) no new maximum was found, the algorithm declares that the peak was found, and continues to the next stage – "Looking for interval".

The method used to detect pop in this stage is determined by the user's input for microwave power – for low power setting the method is Zamir-Berendorf and for high power setting the method is Moving Average.

8.2.3 Looking for Interval

This stage starts after "Looking for Peak" stage. At this point the algorithm knows it has passed the Gaussian peak, and looks to satisfy the manufacturer's recommendation – 2 seconds interval between adjacent pops. This condition was tested and provided good results. The interval search is done using pop detection algorithm, which returns the indices of the found pops, and looking for the correct indices delta. For example, if we sample at 44,100 Hz and look for 2 seconds interval, the algorithm's condition will be delta of 44100*2 = 88,200 samples.

When the desired interval is found, the algorithm declares that the bag is done. As mentions in "Looking for Peak" stage, the detection algorithm is determined by the user's input for power settings.

9 Application

9.1 Guide Lines

The application was developed with the following guide-lines:

- Real-time processing with multi-threading
- Modularity
- Efficiency
- User friendly
- User proof

9.2 Implementation

The application is built for use on Android based devices, as such, it is written in Java while relying heavily on the methodology defined by Android.

The first challenge was recording sound and processing it simultaneously so that no information is lost. Initial implementation used Java Threads but was soon abandoned due to the UI Thread being neglected by the system even with increased priority. The second, and successful, attempt used Android's AsyncTask. Thus, we were able to run a recording thread, a processing thread, and a UI thread.

As this was our first attempt at writing Android software, we knew changes and updates will follow both during work, and deployment, we used the SOLID design principles. Furthermore, the implementation uses a Factory design pattern.

A key characteristic of Android applications intended for the use of the public is user friendliness and user proofing. Guides and instructions were included where seemed important. Furthermore, pitfalls because of mishandling by users were secured, for example, if the application is sent to the background, it will be stopped.

Efficiency was achieved by minimizing calculations and data usage, for example, not a single array with increasing size is used, but rather the array has its data overwritten and thus reused.

9.3 Graphics and UI

9.3.1 Design

The application is designed as a popcorn box with popcorn on top. It uses pattern background for suitability to many screen sizes and resolutions. All buttons and dialog boxes are designed as well with the same design idea. The application's icon is the following:

Figure 12 - Application's icon

9.3.2 Startup Screen

This screen appears when the application starts. Main of the user's actions are performed on this screen. Its main buttons are:

- Starts button – used to move to the detection screen and starts the detection process.
- Power Selection button – opens a dialog for selecting microwave power (by user).
- Power Help button – opens a dialog with explanation about power selection.
- Setting Menu button – opens a popup menu for sound settings, manual and about dialogs.

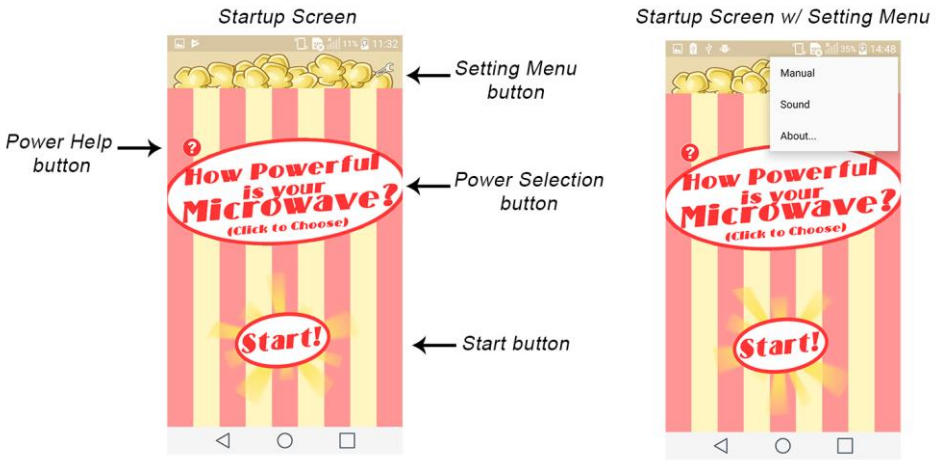


Figure 13 - Startup screen with settings menu and without

9.3.3 Recording and Detection Screen

This screen is presented to the user after "Start button" is clicked. In this screen the recording is started immediately and no user action is required. The main features in this screen are:

- Stage Indication – this area presents to the user the current process flow stage. It contains 3 line for each stage and a popcorn kernel to the left. When a stage is over, the popcorn kernel of the suitable stage is replaced with a popped kernel to indicate that the stage is over. A more detailed picture is presented below.
- Stop button – allowing user to stop the detection operation and go back to Startup activity

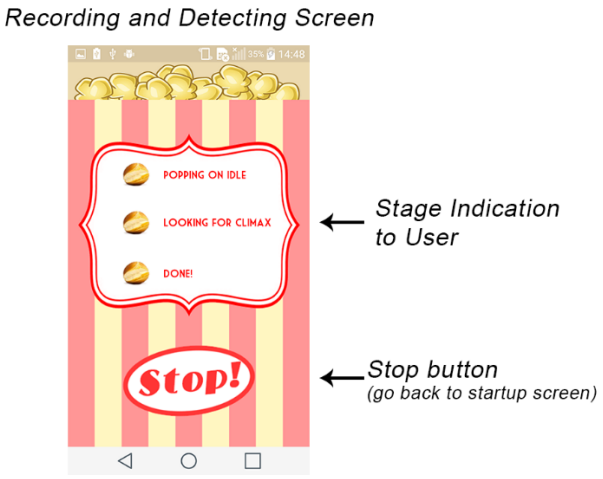


Figure 14 - Recording and Detection screen

Figure 15 - Detection flow on Recording and Detection screen

When the application determine the bag is done, a dialog window pops up. When "OKAY MOVE ON" is clicked the application returns to "Startup screen".

9.3.4 Dialogues

In addition to the main screens, there are several dialogs that play a part in the application's operation:

- Power Selection dialog – This dialog is presented to the user when "Power Selection" button is clicked. It allows the user to change the power setting of the application.

Figure 16 - Power selection dialog

- Sound Selection dialog – This dialog is presented to the user when "Sound" button in settings menu is clicked. It allows the user to change the sound played when a bag is ready.

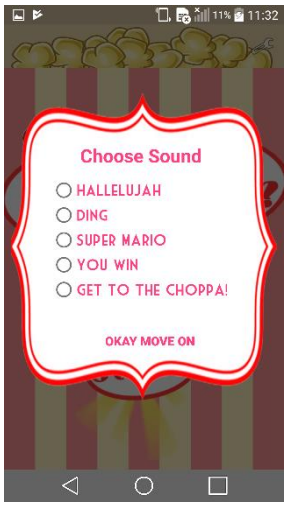


Figure 17 - Sound selection dialog

- Help dialogs – there are 3 help dialogs. First is "Manual", contains a guide for the application's operation. Second is "Power Help" dialog, contains explanation about power settings. Third is "About" dialog, contains version and creators information.

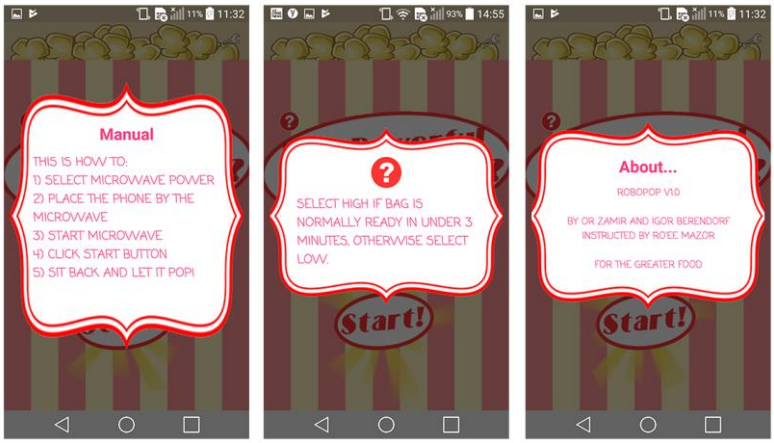


Figure 18 - Information dialogs

Application Results

9.4 General

After constructing a basic application frame we implemented both algorithms Moving Average and Zamir-Berendorf in order to test them in real-life real-time environment. We tested the application by three manners:

- **Playing recordings from data set** and comparing detections from the applications to the known marked recordings.
- **Real-life testing** the application near a working microwave cooking popcorn in **quiet environment**. Quiet environment includes only low background noises such as white noise, microwave noise, dim street noise, etc.
- **Real-life testing** the application near a working microwave cooking popcorn in **noisy environment**. Noisy environment includes talking near microwave, moving chairs and closing doors. This test is trying to simulate the real environment the application is required to deal with.

9.5 Definitions

The test results are presented by several scores, determining with algorithm is best and generally is the application is working as expected. The scores are defines as following:

- Precision – a measure of how much of the detected pops are relevant. Defined as following:

$$Precision = \frac{No.of\ true\ positive}{No.of\ true\ positive + No.of\ false\ positive}$$

Equation 5 - Precision definition

- Recall – a measure of how much of the real pops are detected. Defined as following:

$$Recall = \frac{No.of\ true\ positive}{No.of\ true\ positive + No.of\ false\ negative}$$

Equation 6 - Recall definition

- Weighted absolute distance – a measure of how the detected pops distribution is close to the real pops distribution. The lower the distance, the better is the results.

This measure weights the score by importance – the closer we are to the end of the recording, the results (matching the real distribution) are more important.

$$Weighted\ abs\ distance = \sum_{n} weigths(n) \times Abs(detected\_distribution(n) - (real\_distribution(n))$$

Equation 7 - Weigthed abs distance definition

- Bag success – in real-life testing, a successful bag is a bag with no burned kernels and with a reasonable amount of un-popped kernels left (about 40).

9.6 Results

9.6.1 Recordings from Data Set

The results of this part are divided into two categories. During experimenting the application, we noticed there is a performance difference for each method depending on the microwave cooking power. The difference reflected in all parameters and lead us to the understanding that for different microwave cooking powers may require the use of a different detection method.

Results for high power microwave are presented in the following table:

Method	Recall	Precision*	Weighted Abs Distance
Moving Average	**94.2%**	**98.9%**	**23.7**
Zamir-Berendorf (Derivative)	93.6%	93.2%	38.4

Table 2 - High power microwave application results

Results for low power microwave are presented in the following table:

Method	Recall	Precision*	Weighted Abs Distance
Zamir-Berendorf (Derivative)	**94.9%**	**98.7%**	**56.2**
Moving Average	89.5%	90.5%	81.2

Table 3 - Low power microwave application results

* All precision results ignores double detection of pops.

9.6.2 Discussion

As presented, there was a great difference between the methods for high and low power microwaves. We concluded that each method performs differently for different pops' density. Zamir-Berendorf handles better less dense conditions and Moving Average handles better the high dense conditions. Those differences can be explained by the different

approaches the methods use in order to detect pops – one is adjusting the threshold and one is emphasizing the pops and distinguishes them from the background noise.

Based on those results, we set the application to use different pop detection method for each power setting; for high power the application uses Moving Average; for low power the application uses Zamir-Berendorf.

9.6.3 Real-life Testing

After setting up the application and performed final adjustments, we tested the application under real-life conditions.

For quiet environment, we tested 10 popcorn bags. Out of them 9 were successful bags, with about 40 up-popped kernels left (or less) and a good taste. One bag was over heated and the popcorn was a bit burned, but still eatable. For noisy environment, we tested 5 bags, and 4 of them was successful. Only one bag resulted in a bit burned popcorn.

9.6.4 Discussion

Real-life testing is the final testing of the application. We are happy with the results that indicates that the application is performing well in most cases. However we are aware that the number of tests performed is not sufficient enough to build a reliable statistical view of the application performance. Ideally we could test much higher number of bags for each condition in order to achieve more reliable results.

10 Conclusions

The application was tested in both quiet and noisy environments with good results in terms of kernels popped and quality of popcorn.

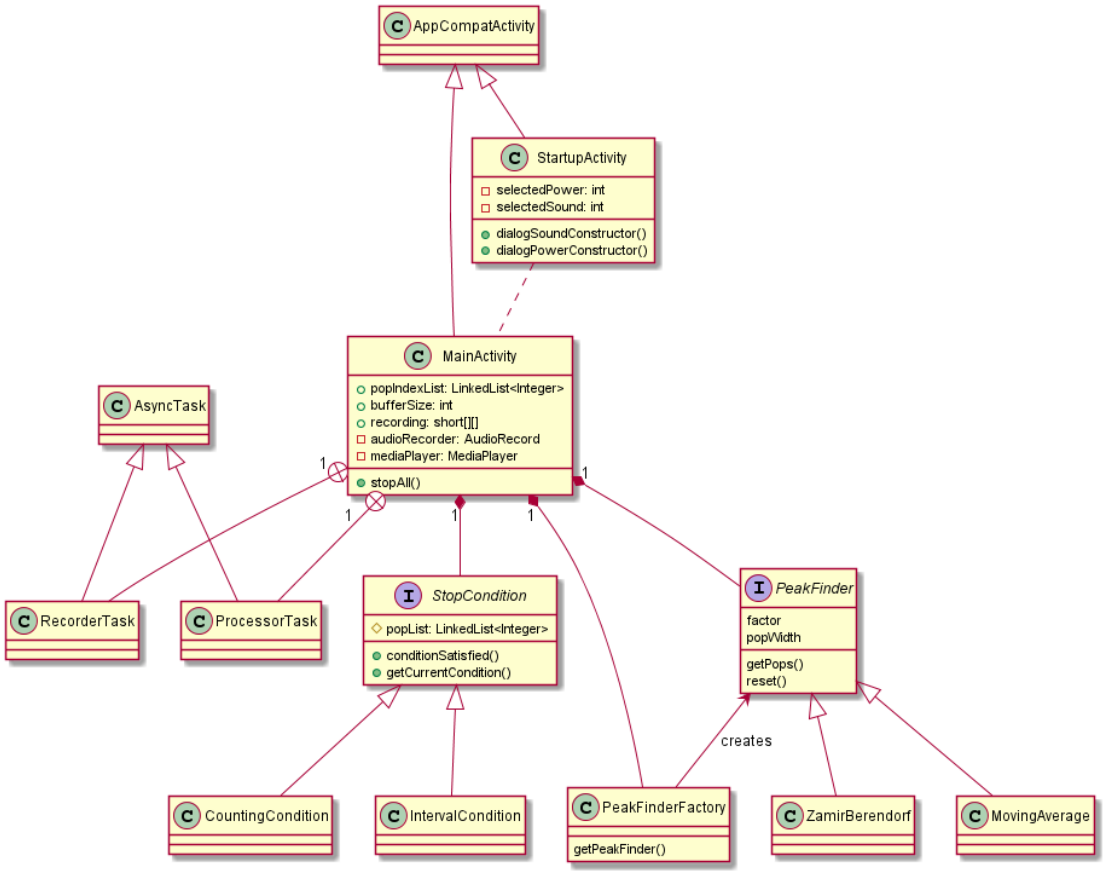
11 Future Development

IoT is most likely the biggest single improvement – the ability to turn off the microwave when the popcorn is ready will enhance the UX and ensure better results. Lacking in hardware this could not have been done by us.

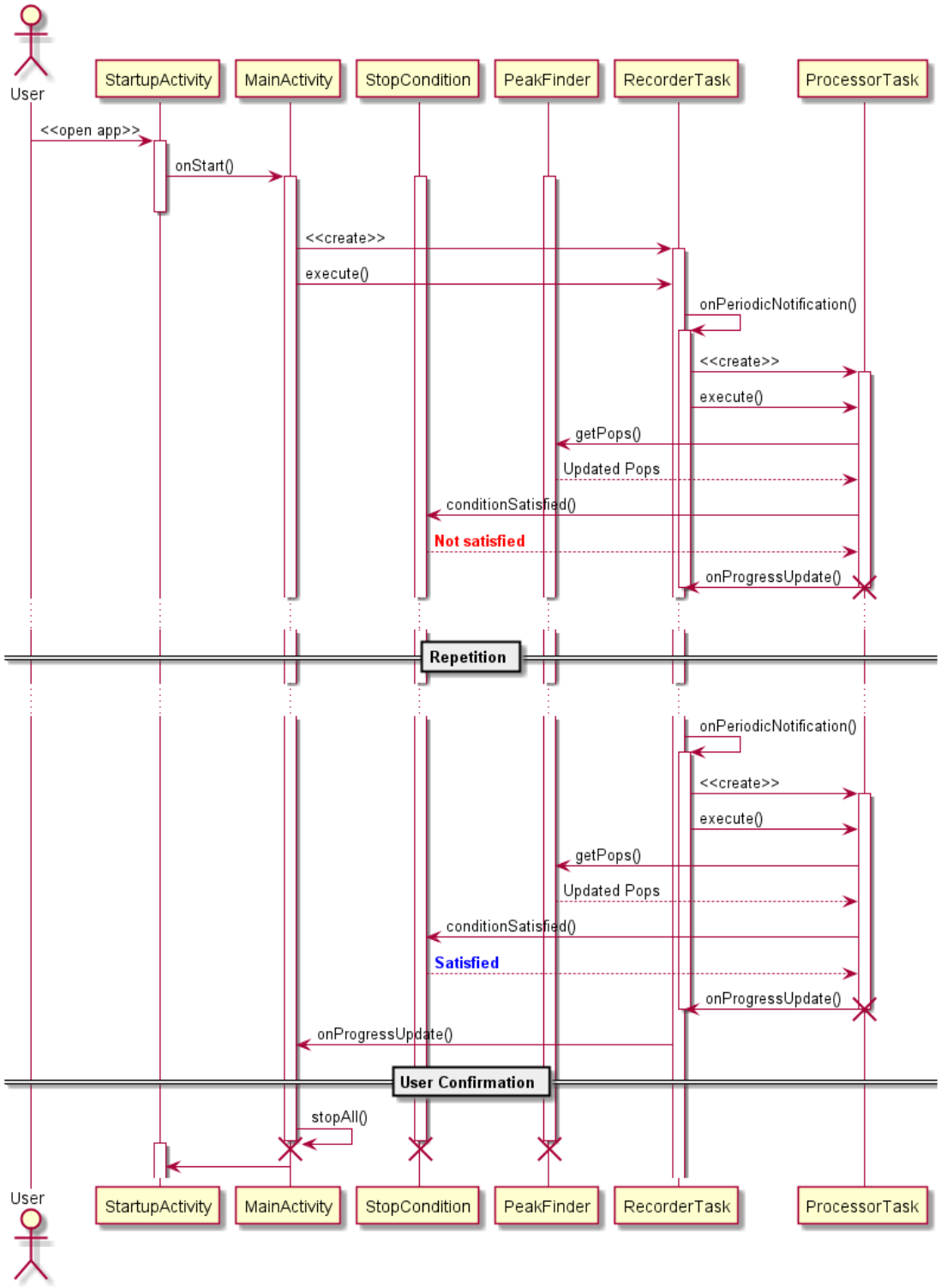
Another possible way to improve the app is making it less susceptible to noise, perhaps by utilizing machine learning. While pursuing this it is important to keep in mind that the application should be lightweight.

12 UML Diagrams

12.1 Class Diagram



12.2 Sequence Diagram



13 Appendix

Link for code file in github - https://github.com/BerIgor/Robopop