Augmented Reality Musical Instruments

Smitesh Sawant, Surabhi Pawar, Prateeksha Singh
Computer Engineering
VCET
Vasai

***Abstract*— We aim to develop image processing software that will allow us to simulate a musical instrument.**

Every laptop comes with a front facing camera. We intend to use this camera to capture live video which will be processed in real-time by the software. The user will act as if he is actually playing the musical instrument, the software will recognize his actions and output appropriate music to the speakers.

We will release a specially designed sheet of paper that has a musical keyboard printed on it. The user has to print this sheet, keep it in front of the camera and start 'pressing' the keys of the printed keyboard, as if it were a real-life keyboard. Our software which runs in the background, tries to simulate the key presses and generate appropriate sound.

I. Introduction

AR Music is a software application which converts a physical action captured in a FOV of a camera to produce a sound. Augmented Reality is a modern technology concept which is defined as, "Augmented reality (AR) is a live direct or indirect view of a physical, real-world environment whose elements are augmented (or supplemented) by computer-generated sensory input such as sound, video, graphics or GPS data. It is related to a more general concept called mediated reality, in which a view of reality is modified (possibly even diminished rather than augmented) by a computer." Combining the technology to augment the reality, the software adds operational renditions of real world sound producing instruments to purport the existence of a virtual instrument which functions as the real one. The software uses basic and advanced image processing algorithms to process the incoming video frames from the camera of a regular laptop or a webcam. Thus it requires a minimum amount of hardware <hardware requirements list>to produce the required result. A printed sheet of a pattern provided as a softcopy with the software is used to partake as a virtual instrument in the augmented reality vision. A bright coloured object (like a pen or a marker) is used as a virtual pointer that is to be tracked and its motion would yield the respective sounds where it lands.

II. Literature Review

There have been numerous projects that have explored the use of augmented reality, three-dimensional interfaces, physical and tangible interaction for generating music. One such project is the 'Augmented Groove' [1]. In the augmented groove, users can play music together, with or without traditional musical instruments, simply by picking and manipulating physical cards on the table. The physical motions of the cards are mapped to changes in musical elements such as timbre,pitch, rhythm, reverb and others. At the same time, users wearing a lightweight head-mounted display (HMD) can see 3D Virtual instruments attached to the cards whose shapes, color and dynamics reflect aspects of the music controlled by the visitors. The music in a sense becomes a physical and tangible object, something we can touch and see as part of our physical environment.

III. Existing Algorithm

The application first calibrates the position of the markers. Each template has a set of dots located on the bordering grids, which are used for calibration. The process of calibration essentially involves calculating the homography from the image observed from the camera to the template image stored on the system. Once calibration is over, the application detects the marker in each input frame. This results in a time series of marker positions, with a few possible gaps due to limitation of the camera. This time series is used to obtain the instants when the marker hits the paper template. The homography calculated during calibration is then used to estimate the location of the marker in the grid by making use of the marker coordinates in the input image at these instants. The music corresponding to the detected position is then played on the speaker.

IV. Implementation

The software used to create this application was MATLAB. Additionally, the vlfeat package is used in order to provide the SIFT algorithm used by the program for template detection . MATLAB provided us with the accessibility to use the basic image processing algorithms. The following are the steps we followed in the application we created:

A. Image Acquisition

Since the project is designed to be low cost, the only required hardware i.e. the laptop itself provides the camera for this stage. The inbuilt webcam serves the purpose of detecting patterns and colors in frames even with the lowest resolution.

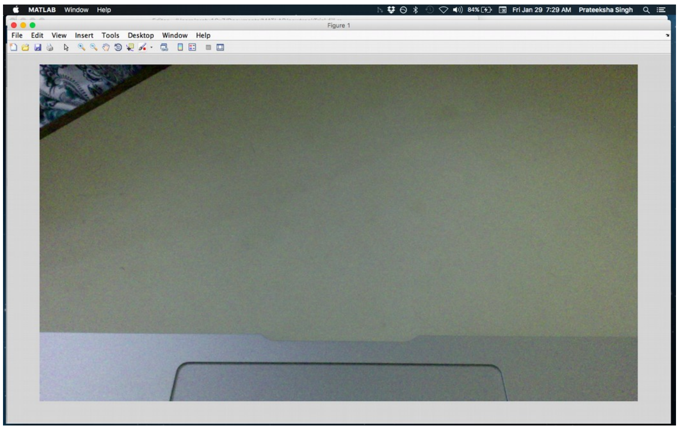

Fig. 2.1.1: Camera test frame

B. *Homography*

Homography (calibration) is a fundamental one-time process carried out in our project. In this step we calibrate the position of the virtual musical instrument i.e. paper which is in front of our camera. For calibration we use an image transform function imported from an external package called vlfeat. The function SIFT(Scale Invariant Feature transform) <add more vlfeat and sift data here> extracts descriptor matrices for a certain number of pixels in the image. The SIFT is applied to both an input frame from the video source and the template image stored in the computer’s memory. These matrices are matched with each other to give a set of pixels on the source image which are completely matched with the corresponding pixels in the template image. This can be observed in a graph plotted in which matches two such images and returns the following result.

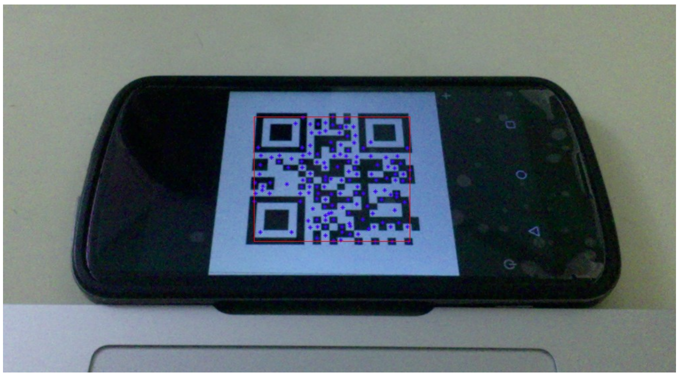

Fig. 2.2.1: SIFT detection

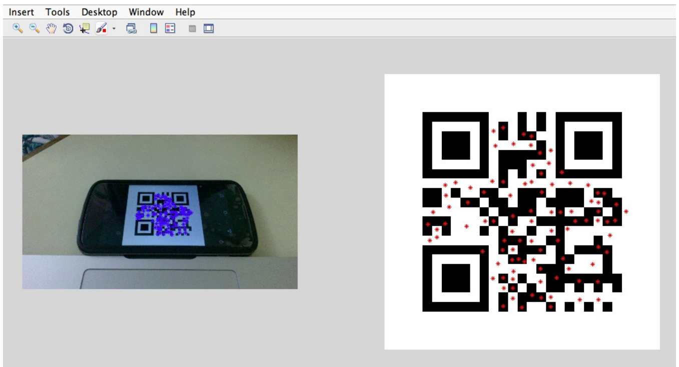

Fig. 2.2.1: Corresponding template features

C. *Object Recognition and Tracking:*

The next step in the implementation is adding the functionality a singular moving object in the input video frames. The object to be tracked is chosen to be of a specific bright colour as we perform colour tracking in our application. The object is tracked and a bounding box is plotted around itself which also marks a centre as a crosshair. The following algorithm is used to carry out object tracking of a red coloured object:

i. START

ii. READ FRAME from the input VIDEO by function getsnapshot(VIDEO)

iii. COPY the RED, GREEN and BLUE channels to separate images.

iv. ASSIGN a threshold image ‘E’ using the values from the RED, BLUE and GREEN channels.

v. SUBTRACT the image E from the original FRAME.

vi. APPLY a median filter with 3x3 matrix to remove any noise.

vii. THRESHOLD the required colour by using the im2bw function.

viii. REMOVE insignificant connected components using BWAREAOPEN function with a calibrated parameter value.

ix. CREATE a Bounding Box on the subtracted and processed image.

x. REPEAT for the next frame.

xi. STOP

<Step by step description of the image can be added>

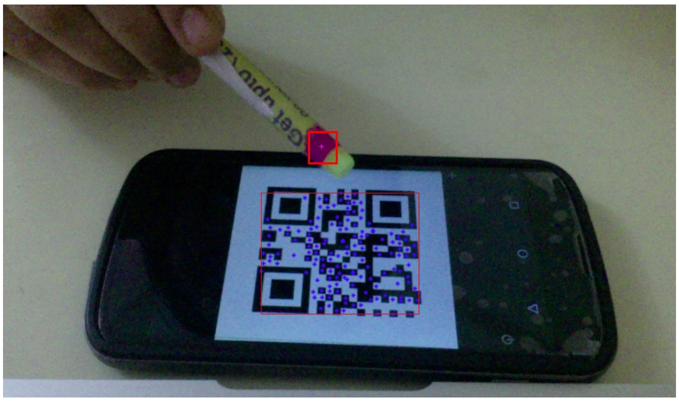

Fig. 2.3.1: Marker tracking overlay

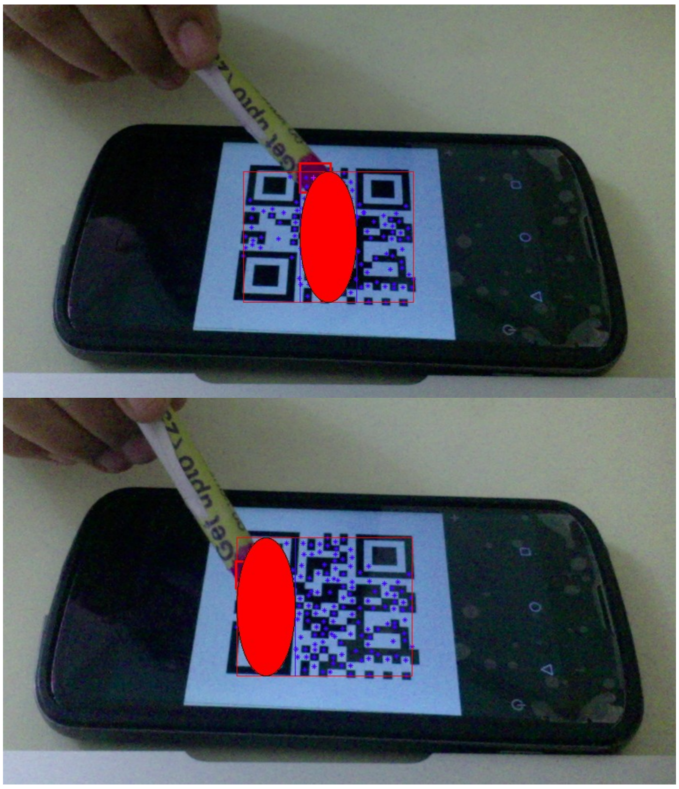

Fig. 2.3.2: Predefined separate areas detected on taps

D. Playing Sound:

The final step in the implementation of the algorithm involves playing the respective sound depending upon the virtual instrument we chose. Every note of the instrument is stored in the system as a WAV file. <more info of WAV file>. MATLAB produces the sound by the use of two separate functions: WAVREAD and SOUND. The WAVREAD function reads a wave file and creates a MATLAB object of the wav file. It stores the signal, the sampling frequency and the number of bits in three different objects which completely define the WAV file. The same three objects are passed as parameters in the SOUND function and the sound is played.

The selection of the particular sound file is triggered by conditional statements which classify the user input to a particular range of pixels. This certain range of pixel would only correspond to a note on a chromatic scale.

V. Results

We have tested the calibration of the template using thresholding and the SIFT algorithm. The program is able to calibrate a printed image of a unique template with respect to a stored file accurately. Further, we have used color tracking from a video input in order to detect the position of the marker. Since the video input taken is low resolution, the processing speed is quite high and the marker position is detected with little to no latency in the output.

The marker position will further enable the corresponding position played on the template to be calculated. adopter communities, with additional ties to the academic and hacker communities.

VI. Discussion

We plan to test our implementation on 3 vidoes made for 3 different instrument templates. Further to test the limits or our application, we have to check its performance when the speed of hitting is increased. Results will depend upon the hit speed. It is expected that at normal hit speeds, we detect all the hits within a distance of one grid. As the speed is increased, the performance may deteriorate. The application should be fairly robust in terms of the quality of light available but not the intensity. Our test videos are made in indoor light whereas we demonstrated our system in outdoor light.

VII. Conclusion

We plan to design, implement and test a virtual musical instrument system. Our system is low-cost and the user needs to print just one sheet of paper for every instrument he/she wishes to play. The latency of the process is small, and the sound output is near real-time, though there is still room for improvement. Our accuracy for the hit localization is 80% for typical user behavior.