# Assisted Music Score Reading Using Fixed-Gaze Head Movement: Empirical Experiment and Design Implications

QINJIE JU, RENÉ CHALON, and STÉPHANE DERRODE, Ecole Centrale de Lyon, France

Eye-tracking has a very strong potential in human computer interaction (HCI) as an input modality, particularly in mobile situations. However, it lacks convenient action triggering methods. In our research, we investigate the combination of eye-tracking and fixed-gaze head movement, which allows us to trigger various commands without using our hands or changing gaze direction. In this instance, we have proposed a new algorithm for fixed-gaze head movement detection using only scene images captured by the scene camera equipped in front of the head-mounted eye-tracker, for the purpose of saving computation time. To test the performance of our fixed-gaze head movement detection algorithm and the acceptance of triggering commands by these movements when the user's hands are occupied by another task, we have designed and developed an experimental application known as EyeMusic. The EyeMusic system is a music reading system, which can play the notes of a measure in a music score that the user does not understand. By making a voluntary head movement when fixing his/her gaze on the same point of a music score, the user can obtain the desired audio feedback. The design, development and usability testing of the first prototype for this application are presented in this paper. The usability of our application is confirmed by the experimental results, as 85% of participants were able to use all the head movements we implemented in the prototype. The average success rate of this application is 70%, which is partly influenced by the performance of the eye-tracker we use. The performance of our fixed-gaze head movement detection algorithm is 85%, and there were no significant differences between the performance of each head movement.

CCS Concepts: • **Human-centered computing → Human computer interaction (HCI)**; *HCI design and evaluation methods*; *Interaction techniques*; Usability testing; Auditory feedback; Pointing; Gestural input;

Additional Key Words and Phrases: Eye-tracking, gaze interaction, head movement, music score.

**ACM Reference Format:**
Qinjie JU, René CHALON, and Stéphane DERRODE. 2018. Assisted Music Score Reading Using Fixed-Gaze Head Movement: Empirical Experiment and Design Implications. 1, 1 (October 2018), 29 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

Eye-tracking is a technology which aims at providing an estimate of gaze direction by recording eye movements. The equipment used to provide gaze direction information, and possibly the focused object, is called an eye-tracker or oculometer [7]. Thanks to the remarkable progress made by computer power and the performance of eye-trackers, a renewed interest in eye-tracking has been observed in the HCI (human computer interaction) community as an input modality to facilitate different types of interactions.

Authors' address: Qinjie JU; René CHALON; Stéphane DERRODE, Ecole Centrale de Lyon, 36 Avenue Guy de Collongue, LYON-Ecully, 69134, France, qinjie.ju@doctorant.ec-lyon.fr, rene.chalon@ec-lyon.fr, stephane.derrode@ec-lyon.fr.


XXXX-XXXX/2018/10-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

In our research, we seek to study the potentialities of eye-tracking to interact with objects in the real world while leaving both hands free to perform gestures related to the user's main activity. One major problem for applying eye-tracking in the field of HCI is that this technology lacks a convenient action triggering method. By comparing the advantages and drawbacks of different solutions proposed by the researchers (see section 2.2), we decided to investigate the combination of eye-tracking and fixed-gaze head movement (voluntary head movement with gaze fixed on the same point), since this enables us to trigger various commands without using our hands or changing gaze direction. Our research question is to propose an algorithm for fixed-gaze head movement detection using only a head-mounted eye-tracker equipped with a scene camera, to test the performance of this algorithm, and to assess the fixed-gaze head movement detection technique in an experimental application.

Our approach is mainly experimental by the development of application prototypes implementing these concepts and their evaluations through empirical studies. The application that we present in this paper is related to the learning of music and, more particularly, to the learning of a musical instrument. This application is intended to help beginners of music interact with music scores quickly and conveniently while keeping both of their hands free to practice the instrument.

Our main contributions are:

- We have proposed a new algorithm for fixed-gaze head movement detection using only scene images with gaze point overlay provided by a head mounted eye-tracker, which may be more efficient and which allows us to interact with objects in the environment.
- We have evaluated the performance of our algorithm and the feasibility of this approach through an intensive usability test.
- We have highlighted a use case in which eye-tracking provides real added value compared with other interaction devices.

The remainder of the paper is organized as follows. We propose in section 2 a state-of-the-art of eye-tracking applications in HCI, particularly for interactions with physical objects in the surroundings and in mobile situations. The techniques used to trigger commands are also mentioned and compared in this section. In this context, we present in section 3 the design and development of our experimental application for interactive music reading known as EyeMusic. Especially, details about the fixed-gaze head movement detection algorithm are also provided in this section. Section 4 describes the setup and procedure of the usability test we conducted on 41 participants in order to evaluate the performance of the first prototype of this application, while the experimental results are described and analyzed in section 5. Section 6 concludes the paper with conceivable future work.

## 2 STATE-OF-THE-ART

### 2.1 Mobile and Ubiquitous Interaction by Gaze

Research into the smart environment began in the late 1990s, and yielded new possibilities for the application of eye-tracking in the field of HCI. Several studies are based on using eye-tracking to select a device and trigger interactions with it by placing cameras next to each device [28, 37]. According to the experiments, the majority of users quickly get into the habit of looking at the device with which they will make interactions [17, 24]. Also, this does not require extra effort on the part of the users, allowing them to then interact remotely with this device more quickly.

Head-mounted eye-trackers with a front camera able to capture scene images, are also used for the benefit of interactions with the physical environment. This is because they can act as a selection tool without requiring multiple cameras placed in the environment. Thanks to pattern recognition methods, researchers have succeeded in detecting and recognizing the viewed object

by calculating the correspondence between the real-time scene image and a pre-known device with images recorded in a database beforehand [2, 27, 29].

As the head-mounted eye-trackers became increasingly mobile and robust, the application field of eye-tracking was no longer restricted to a single room. Researchers have considered applying eye-tracking outdoors, in order to perform interactions based on the user's position and gaze with the help of computer vision or head orientation detection tools [1, 10].

In addition, with a camera filming the watched scene, it becomes possible to provide users with assistance in interpreting what they cannot understand. By applying this in the context of the museum, Toyama et al. have created a Museum Guide 2.0 application that can provide the user with additional information on the exhibit of interest [33]. Kobayashi et al. have worked on the combination of an eye-tracker and a character recognition system by carrying out a recognition device that applies the OCR (Optical Character Recognition) process around the region pointed by the user's gaze: this makes it possible to give users complementary information on the recognized words in multiple forms (translation, image, etc.) [14]. The EyeMusic application we present in this paper deals with this. We propose to provide audio feedback corresponding to the fixed measure on the music score to assist novice musicians when they need to learn an unfamiliar melody, with the help of a head-mounted eye-tracker equipped with a scene camera.

### 2.2 Problems in the Use of Eye-tracking for HCIs and Proposed Solutions

Although the use of eye-tracking in HCI provides many benefits, it still poses a number of problems [4]:

- Gaze fixation often lacks precision: traditional input methods such as the mouse attain the accuracy of a pixel. On the other hand, even with the most accurate eye-tracker nowadays, gaze fixation can generate a deviation of 0.5 degrees due to the fovea surface [38].
- Lack of action triggering method: generally, the selection of an element by a traditional interface is finished by a mouse click. However, this action is not easily achievable by the eyes. Simulating a mouse click by blinking your eyes is not a favorable solution, since it is difficult to differentiate a natural blink from a voluntary one. Furthermore, eyelid blinking can result in a downward drift of the gaze and a temporary gaze data absence. Another alternative is to simulate the mouse click by a dwell time [12]. However, this method is very time consuming (600-1000ms [18]), thus reducing the potential gains of eye speed.
- The eyes are also used to inspect the scene: the main task of eyes is to observe information in the environment. However, in the case of eye-tracking, it is not obvious to know whether the user wants to simply observe an element or to interact with it. This problem, identified by Jacob, is called the Midas touch problem [13].

Researchers have proposed several solutions to solve one or more of these problems. Instead of fixing the gaze on the button to click, voluntary eye movements are studied in order to pass various commands [3, 5, 22]. In this case, gaze point accuracy is no longer important, but these movements can cause eyestrain. For the same purpose, Esteves et al. have worked on the use of smooth pursuit to select target objects on a screen [8] or in the physical environment [36]. As the smooth pursuit of eyes is not used to observe the environment, this avoids the Midas touch problem.

Apart from the use of smooth pursuit and voluntary eye movements, the majority of the solutions proposed earlier are based on coupling the gaze with another device. This secondary device can stem from a traditional input method (such as a mouse [6, 38] or a touch pad [20, 31, 34]) or a new technology (such as the brain-computer interface [32] or motion sensing devices [35]). Recently, 3D (three-dimensional) gestures have been adopted in the field of video games as an input modality, arousing the interest of more and more commercial systems. This is thus also a potential modality

to be coupled with eye-tracking. By combination with a 3D gesture detection system, researchers have managed to select and manipulate objects remotely, by gaze and 3D hand gestures, at greater speed [11, 35]. The fixed-gaze head movement, which combines head movements with gaze, has also been studied to interact with a screen [19, 23, 30].

We have compared the proposed modalities in multiple aspects. The advantages and drawbacks of each modality extracted from the papers are shown in Table 1.

Table 1. Comparison of different modalities for triggering commands in combination with a head-mounted eye-tracker.

| | Requirement of an extra device | Need to release the hands | Need to change gaze direction | Ability to trigger numerous commands | Command execution time |
|---|---|---|---|---|---|
| Fixed-gaze head movements | No | No | No | Medium | Medium |
| Voluntary eye movements | No | No | Yes | Good | Medium |
| Smooth pursuit | Yes | No | Yes | Medium | Medium |
| Mouse/ touch pad | Yes | Yes | Yes | Medium | Short |
| Hand/arm 3D movements | Yes | Yes | No | Good | Medium |
| Foot movements (pedal) | Yes | No | No | Poor | Short |
| Voice control | Yes | No | No | Good | Medium to Long |

It can be seen that voluntary eye movements and smooth pursuit require a change in gaze direction. In that case, the gaze can no longer serve as a pointing modality like the cursor on a screen. While there are multiple objects in the scene, they can be used to either select an object or trigger commands on a selected object, instead of selecting an object and triggering commands on it at the same time. The two-dimensional (mouse/touch pad) or three-dimensional hand movements, on the other hand, can be used to trigger commands on the fixed object. However, as they require the users to release their hands in order to perform these actions, they are not convenient if both of the users' hands are occupied by another task. In addition, the mouse or touch pad may require the user to look at it in order to put his/her hand on it, which would disturb the user. Therefore, the fixed-gaze head movement, foot movements, and voice control are rather competitive if we want to trigger commands on the fixed object while keeping both hands free. In these three modalities, the ability to trigger numerous foot movement commands is rather poor. Voice control is capable of triggering a large number of commands, but its execution time when triggering commands may be longer than the other modalities since the user needs to pronounce these commands. The latter can be chosen according to application needs. When we only need a limited number of commands

and we do not want them to be time-consuming (like the EyeMusic application), the fixed-gaze head movement would be the better choice. However, if we need to trigger a greater number of commands, voice control would be a potential choice.

### 2.3 Fixed-gaze Head Movement Detection for Triggering Commands

The combination of head movements and gaze using only an eye-tracker has been studied since 2012. Mardanbegi et al. have tried to detect fixed-gaze eye movements only with a head-mounted eye-tracker, and proposed to use this kind of movement to interact with a screen [19]. Similarly, Spakov et al. have studied the same technology using a remote eye-tracker [30]. The work of Nukarinen et al. is not limited to triggering a simple action, as they investigated performance optimization of head movements in complex and continuous interactions [23]. This research makes it possible to couple gaze with head movements without the need for an extra device (apart from the eye-tracker). Since both foot movements and voice control require an extra device (a pedal or a microphone), the advantage of using fixed-gaze head movement in combination with eye-tracking to trigger commands is further highlighted.

To the best of our knowledge, all the researchers have investigated detection of fixed-gaze head movements with the help of eye images captured by eye-tracker: some of them use head-mounted eye-trackers [19], while others use remote eye-trackers [23, 30]. On the contrary, in our project we tried to detect fixed-gaze head movements from the scene images captured by the scene camera of the head-mounted eye-tracker, for the purpose of saving computation time. Moreover, the application field of the previous works is limited to interacting with a screen, whereas we attempt to interact with any object in the environment.

## 3 EYEMUSIC: INTERACTIVE LEARNING OF MUSIC

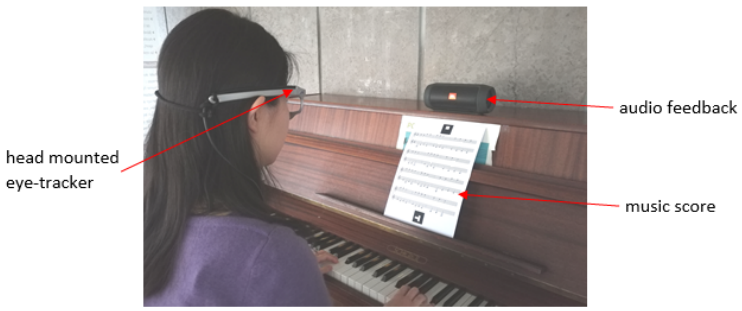



Fig. 1. Typical configuration of EyeMusic in learning to play the piano.

Learning music provides many benefits, not only intellectual but also personal and cultural [15]. Music reading is essential for musicians, and some time is required to develop the necessary skills. Learning to read music scores is no simple task for beginners: it is even more complicated than learning a new language, since language is one-dimensional while music is two-dimensional (height and length of notes). Consequently, the creation of a music learning support system can be very significative.

Normally, when a beginner learns to play a melody with a musical instrument, he/she often needs to imitate the performance of another musician as he/she has not yet mastered music reading skills. In order to hear the notes of a specific measure, he/she has to put aside his/her musical instrument, play the corresponding audio file and then look for the right temporal position. This

process is not convenient for novice musicians and is very time consuming. The EyeMusic system implemented in this project is a music learning system that can play the notes of a measure in a music score that the user does not understand, with the help of a head-mounted eye-tracker. When the user encounters difficulties in music score reading, by a simple head movement, he/she will be able to trigger the audio feedback of the measure fixed with his/her gaze without releasing his/her hands occupied by the musical instrument (flute, guitar, piano, etc.). An example of learning to play the piano with the help of EyeMusic is shown in Figure 1.

### 3.1 Head Mounted Eye-Tracker

There are 2 main types of eye-tracker: the stationary eye-tracker and the head-mounted eye-tracker. For this application, we have chosen to use the head-mounted eye-tracker equipped with a scene camera. This is because, apart from calculated gaze information, this kind of eye-tracker can also provide us with scene images that are useful for music score recognition and head movement detection.

The device we actually use is the Eye Tracking Glasses 2 Wireless produced by SMI (SensoMotoric Instruments)[1]. This is a head-mounted eye-tracker in the form of a pair of glasses. Two internal cameras film eye movements at a sampling rate of 30Hz in order to calculate gaze direction, while a third camera, located in front of the glasses, captures scene images at a frame rate of 24fps (frames per second). Thanks to the software development kit of this equipment, we have direct access to precise gaze information and scene images.

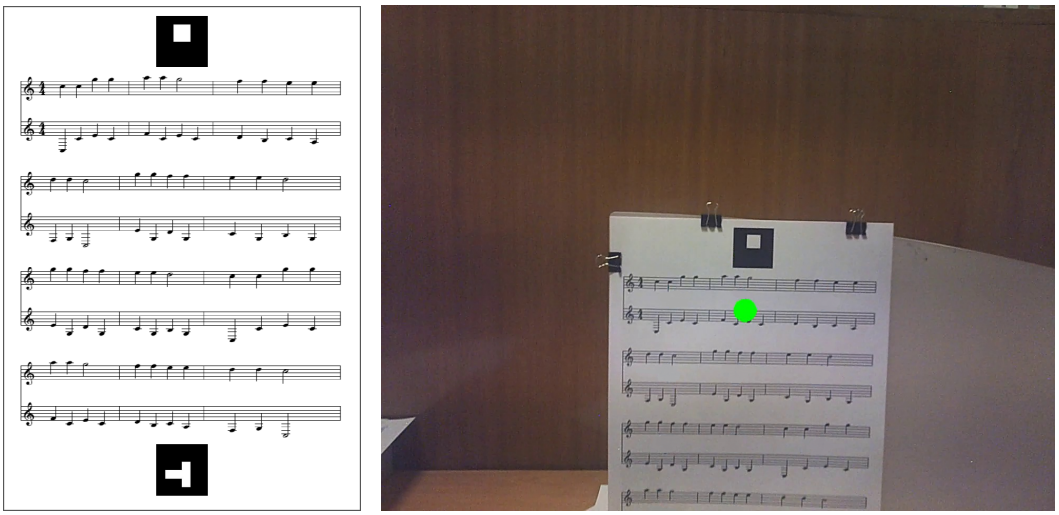
### 3.2 Notes and Score Recognition

To recognize the notes fixed by the user, the most direct way is to apply optical music recognition (OMR) [21, 26] to the scene image. However, as music is more complicated than written language, the performance of the OMR algorithms used in commercial software and current research is not satisfactory enough. Alternatively, we decided to link each physical music score with its digital version (MIDI file) with the help of different markers and to use the correspondence between the location on the page and the time position in the MIDI file to obtain the corresponding notes in the desired measure. In this prototype, we have chosen to use ArUco markers [9], which are quite numerous and easily detectable using functions in an image processing library. An example of a music score attached with ArUco markers is shown in Figure 2a. In our application, 2 markers are attached on each page of the music score (1 at the top, 1 at the bottom) to make sure that there is at least one marker visible on the scene image when the user is reading the score. This is because, due to the limited view angle of the scene camera, a part of the music score may be outside the scene image as in Figure 2b. Each set of markers is used to identify 3 different things: the unique music score that is being read, which page of the music score is being read, and the relative position of each measure on this page.

### 3.3 Relative Gaze Position Calculation

While the gaze position we obtain from the eye-tracker takes the form of coordinates on the scene image, in fact what is useful in our application is the relative gaze position with respect to the music score. In that case, in order to know which measure in the music score interests the user, we compute the homography matrix using the coordinates of detected ArUco markers. The homography matrix is a 3*3 matrix that relates the coordinates transformation between 2 planar surfaces in the field of computer vision. In our case, with the help of the homography matrix, we

[1]http://www.smivision.com/wp-content/uploads/2017/05/smi_prod_ETG_120Hz_asgm.pdf



(a) (b)

Fig. 2. Example of a music score and its view on the scene image: (a) example of a music score with 2 ArUco markers, (b) example of a scene image with gaze point overlay while part of the music score is outside the view angle of the scene camera.

are able to transform gaze coordinates in pixels on the scene image into relative gaze coordinates with respect to the sheet of paper (music score).

The gaze direction obtained via the API (application programming interface) provided by SMI has a sampling rate of 30Hz, but the scene camera frame rate is 24 fps. We need to find a solution to obtain the gaze information corresponding to each frame. As the software of the SMI eye-tracker automatically adds a green point on each scene image (like the one on the music score in Figure 2b) to imply the corresponding gaze position, we decided to detect the green point to obtain gaze coordinates on each scene image. This can also be easily achieved using functions in an image processing library. Constrained by the computer's computing power and the performance of our algorithm, the relative gaze coordinates are calculated every 2 frames: thus computing is 12fps.

### 3.4 Fixed-Gaze Head Movement

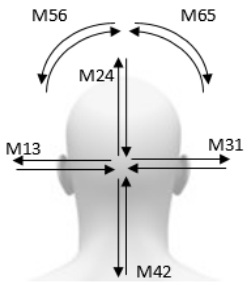
The music score reading process is accomplished through fixations and saccades. If the music score is single-row, there tends to be a series of horizontal saccades followed by a vertical movement and then more horizontal saccades. When the music score is double-row, the musicians tend to use vertical movements alternately from one half of the staff to the other half, along with the horizontal saccades [25]. Natural head movements during the music score reading process are always accompanied by variations in gaze direction. On the other hand, thanks to the vestibulo-ocular reflexes of eyes, which compensates head movement by an eye movement in the opposite direction, a voluntary head movement with gaze fixing on a precise point is possible.

We choose to trigger commands by fixed-gaze head movements in the EyeMusic application because this kind of movement has several obvious advantages in this context:

(1) Fixed-gaze head movements can be detected from scene image variation and change of gaze point: no extra equipment (except the head-mounted eye-tracker) is required in this system.
(2) Fixed-gaze head movements can be performed quickly without releasing the hands.

(3) During fixed-gaze head movements, the user's gaze point does not change, so it would not disrupt the user's reading process.
(4) Compared with natural head movements that we use all the time, fixed-gaze head movements are not often used during the reading process, although some people have the habit of beating time by nodding. We can thus avoid the Midas touch problem.

There are many possible head movements. In this project, we investigate detection of movements combined by basic actions, and return to the natural position: as these movements are easy to make, and as the head is in the natural position after each movement, it is convenient to continue reading or to make another movement afterwards. Figure 3 shows the 6 movements dealt with in our application.




| | |
|---|---|
| M13 | Head turns left and right to the natural position |
| M31 | Head turns right and left to the natural position |
| M24 | Head up and down to the natural position |
| M42 | Head down and up to the natural position |
| M56 | Head tilts left and right to the natural position |
| M65 | Head tilts right and left to the natural position |

Fig. 3. Basic head movements used in EyeMusic.

### 3.5 Fixed-Gaze Head Movement Recognition Techniques

A fixed-gaze head movement should comply with 2 conditions: The first one is that the gaze is fixed at the same position, the second is that there is a desired head movement. When relative gaze coordinates are computed, it is easy to detect whether or not the gaze point is fixed on the same measure during a head movement. As the eyes move in the opposite direction and with the same speed as the head during fixed-gaze head movements, it is possible to measure head movements through eye movements. Figure 4 illustrates eye movement during fixed-gaze head movement. Mardanbegi et al. investigated detection of eye movements using eye images captured by eye cameras [19]. In this project, we investigate detection of fixed-gaze head movements using only scene images with gaze point overlay in order to save computation time, especially for vertical and horizontal movements.

We use two different algorithms in parallel to detect the 6 fixed-gaze head movements we have chosen for this application: one for vertical and horizontal movements (M13, M31, M24, and M42), the other for rotational movements (M56 and M65).

*3.5.1 Vertical and horizontal movement recognition.* When making vertical or horizontal fixed-gaze head movements, the user's eyes move vertically or horizontally, and the gaze point coordinates on the scene image also change. Thus, M13, M31, M24 and M42 are identified by a variation in the derivatives of gaze point coordinates. For example, when the user makes the M24 movement with his/her gaze fixed on the same point, the coordinate y of his/her gaze point should have a variation as in Figure 5a. This kind of coordinate variation can be detected by calculating its derivative: if the derivative of coordinate y possesses in turn a positive peak and a negative peak, which have crossed the thresholds within a limited time as in Figure 5b, we admit that there is a required eye movement. The thresholds were chosen through preliminary tests with our lab colleagues. During these tests,

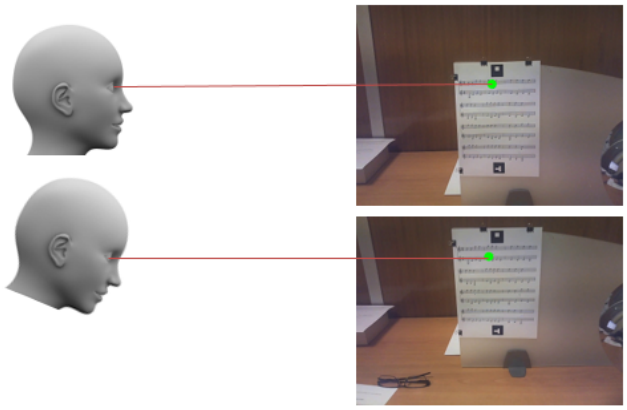


Fig. 4. Gaze point variation during vertical fixed-gaze head movements.

we asked our lab colleagues to make fixed-gaze head movements quickly and comfortably as we do not want our approach to be time consuming or tiring. We found that the derivative of gaze coordinates always possesses peaks (one negative peak and one positive peak) with an absolute value higher than 15 pixels every 1/12 second (180 pixels/second since we process 12 frames per second) when making horizontal or vertical fixed-gaze head movements. This implies that the highest angular speed can be greater than 10 degrees per second. Also, the duration constraint is set by requiring the two peaks to occur within 10 processed frames (0.8s), as it is not very time consuming. Indeed, our preliminary tests show that the quick fixed-gaze head movement satisfying this constraint is feasible for all participants.

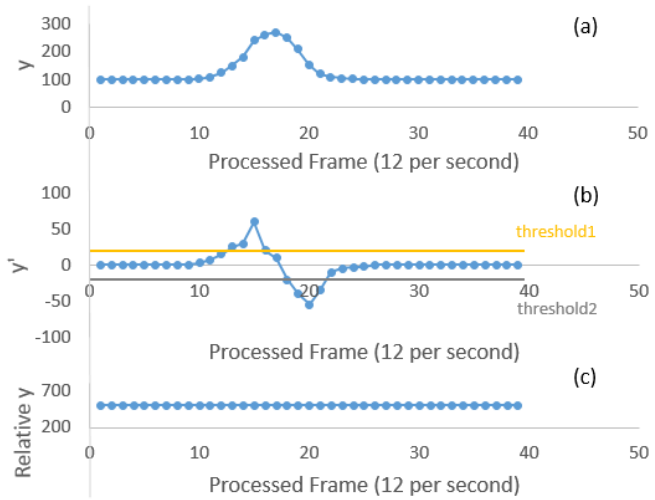



Fig. 5. Variations in gaze coordinates during fixed-gaze M24: (a) variation in y coordinate, (b) derivative of y coordinate, (c) variation in relative y coordinate.

Ideally, if the relative coordinates of the gaze point on the music score remain at the same point or change slightly in the region of the same measure during the required eye movement as in Figure

5c, we can be sure that the user has made a fixed-gaze head movement. Unfortunately, during head movements, the sharpness of scene images is not guaranteed. There are thus several frames without detectable ArUco markers, meaning that we lose relative gaze information during the movements. Figure 6a shows the relative y coordinate signal that we actually obtain during a fixed-gaze M24: the part covered by a gray square is the period during which we may lose relative gaze coordinates. As a result, in the worst case, we only obtain relative gaze coordinates at the beginning and at the end of each movement. Consequently, we seek to distinguish different situations which may cause an eye movement like this and exclude the other cases.

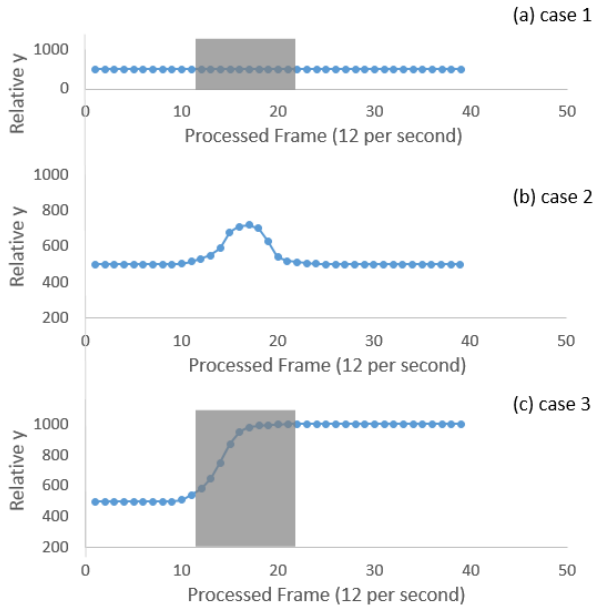



Fig. 6. Differences on the relative y coordinate among the 3 cases, which may cause a desired variation in gaze coordinates: (a) relative y coordinate during fixed-gaze head movement, (b) relative y coordinate during pure eye movement, (c) relative y coordinate while the user's eyes and head move at the same time.

Normally, an eye movement such as shown in Figure 5a occurs in 3 different situations:

Case 1: Fixed-gaze head movement,

Case 2: Natural eye movement while the head does not move,

Case 3: The user's eyes and head move at the same time in the same direction and the eyes move faster than the head.

Case 1 is the fixed-gaze head movement that we want to detect through the derivatives of gaze coordinates: thus case 2 and case 3 must be excluded.

Case 2 exclusion: For the second case, the relative y coordinate signal should resemble what is shown in Figure 6b if the variation in coordinate y is like the signal in Figure 5a. During natural eye movement, there is a significant variation in relative gaze coordinates compared with the variation in gaze coordinates on the scene image. Also, while these two variations follow the same direction, there is no problem of loss of gaze information as the user's head does not move. Thus, natural pure eye movements can be excluded by setting a constraint on the ratio between the derivative of relative gaze coordinates and the derivative of gaze coordinates. We choose to set a threshold on the ratio of the two derivatives rather than setting a constraint on the variation in relative gaze

coordinates, as the latter can only exclude huge eye movements while the former is also capable of eliminating small eye movements and tremors. The ratio we set in our prototype is 3.0, which is equivalent to the ratio between the variation in relative gaze coordinates and the variation in gaze coordinates when the music score is 35cm (minimal distance for reading) from the user. Normally, the distance between the user's eyes and the music score should be greater than 35cm, so the ratio should be larger than 3.0 when making pure eye movements. During horizontal and vertical fixed-gaze head movements (case 1), this ratio should either not exist (due to the lack of sharpness of the scene image) or be smaller than 3.0 (since the gaze stays at the same position): this has been verified by the preliminary tests with our lab colleagues. Consequently, this threshold can be used to exclude pure natural eye movements without ignoring the desired fixed-gaze head movements.

Case 3 exclusion: For the third case, there is a huge difference between relative gaze coordinates before and after the movement, as is shown in Figure 6c, and the gaze point is fixed on a different measure after the movement. Although we may lose relative gaze information during the movement, it is easy for us to differentiate it from fixed-gaze head movements during which relative gaze coordinates are stable before and after the movement.

To conclude, if there are 2 peaks (a positive peak and a negative peak) on the derivative of coordinate x (or y) which have crossed 15 (THRESHOLD_PEAK) within 10 processed scene images (THRESHOLD_DURATION), during which the ratio between the derivative of relative coordinate x (or y) and the derivative of coordinate x (or y) does not exceed 3.0 (THRESHOLD_RATIO), and the gaze is fixed on the same measure before and after the start of the movement (first peak), we admit that there is a vertical (or horizontal) fixed-gaze head movement. While we separate the conditions for vertical and horizontal movements in the algorithm below in order to make it clearer, in our prototype these movements are detected at the same time.

**Algorithm 1: Vertical and horizontal fixed-gaze head movement recognition**

```
def vertical and horizontal fixed-gaze head movement recognition                    vertical movement
    (processed frame number n)                                                        recognition
    get gaze coordinates → x(n), y(n)
    calculate derivatives → x'(n), y'(n)
    isSharpEnough = FALSE
    if ArUco marker detected
        isSharpEnough = TRUE
        get marker coordinates
        calculate relative gaze position → xR(n), yR(n)
        calculate derivatives of relative gaze coordinates → x'R(n), y'R(n)
        search the corresponding measure number → nMeasure(n)
    if x'(n)> THRESHOLD_PEAK (or < -THRESHOLD_PEAK)                                    y'(n)
        if isSharpEnough = FALSE or x'R(n)/ x'(n) < THRESHOLD_RATIO                  y'R(n)/ y'(n)
            peak_x(n) = 1 (or -1)                                                     peak_y(n)
            for i = 0 to THRESHOLD_DURATION
                if peak_x(n-i) = -1 (or 1)                                            peak_y(n-i)
                    for j = 0 to i
                        if nMeasure(n- THRESHOLD_DURATION) = nMeasure(n-j)
                            return M13 (or M31) detected                              M24 (or M42)
    return 0
```

*3.5.2 Head tilt recognition.* On the other hand, head tilt cannot be detected from the variation in gaze point coordinates on the scene image, as the variation in gaze coordinates is linked to the gaze direction when the user's head is in the natural position (without making head movements). There may be huge variations in x and y coordinates at the same time if the gaze is fixed on the corner of the scene image as in Figure 7, but if the gaze is fixed on a point near the image center the coordinates may be nearly stable. Therefore, instead of calculating the derivatives of gaze coordinates, we detect head tilts by scene image processing. With the homography matrix calculated in section 3.3, we can easily obtain the angle of rotation in 3 dimensions for each frame using openCV functions. We use 2 thresholds to identify head tilts from a series of rotation angles: the first is on the derivative of rotation angle z (axis perpendicular to the image plane), while the second constraint is on the maximum absolute value of the difference between the rotation angle z during the head tilt and the rotation angle z before the head tilt. We use these two limits simultaneously in order to eliminate tiny variations in rotation angle z, which may be caused by the image processing deviation and to exclude the case that the user's head remains at a position with a significant rotation angle for some unknown purpose (for example the user's head may incline to the left when playing the violin). These thresholds were chosen through preliminary tests with our lab colleagues, and the duration constraint is the same as for horizontal and vertical head movements (10 processed frames).

The algorithm of fixed-gaze head tilt recognition is shown below. If there is a positive peak and a negative peak on the derivative of rotation angle z which has crossed 1.9 (THRESHOLD_ROTATION) within 10 processed frames (THRESHOLD_DURATION), and the maximum absolute value of the difference between the rotation angle z value before the head tilt and during this movement is greater than 10 degrees (THRESHOLD_DIFFERENCE), we admit that there is a desired head tilt.

**Algorithm 2: Fixed-gaze head tilt recognition**

```
def fixed-gaze head tilt recognition (processed frame number n)
    calculate the corresponding measure number → nMeasure(n)
    calculate the rotation angle z → rZ(n)
    calculate derivative of rotation angle z → r'Z(n)
    potentialHeadTilt = FALSE
    if r'Z(n)> THRESHOLD_ROTATION (or < -THRESHOLD_ROTATION)
        peak_z(n) = 1 (or -1)
        for i = 0 to THRESHOLD_DURATION
            if peak_z(n-i) = -1 (or 1)
                for j = 0 to i
                        if |rZ(n-j) - rZ(n-THRESHOLD_DURATION) | > THRESHOLD_DIFFERENCE
                            potentialHeadTilt = TRUE
    if potentialHeadTilt = TRUE
        for i = 1 to THRESHOLD_DURATION
            if nMeasure(n-i) ≠ nMeasure(n)
                return 0
        return M56 (or M65) detected
    return 0
```

*3.5.3 Global algorithm.* As head tilts may also cause variations in gaze coordinates, we identify head tilts first. Variations in gaze coordinates are examined to detect horizontal and vertical head

movements only if there is no head tilt in progress according to the variation in rotation angle z. The global algorithm is shown below.

**Algorithm 3: Global algorithm**

```
for each processed frame (number n)
    run fixed-gaze head tilt recognition algorithm (n)
    for i = 0 to THRESHOLD_DURATION
        if peak_z(n-i) = -1 (or 1)
            for j = 0 to i
                if |rZ(n-j) - rZ(n-THRESHOLD_DURATION) | > THRESHOLD_DIFFERENCE/2
                    goto next frame
    run vertical and horizontal fixed-gaze head movement recognition algorithm (n)
```

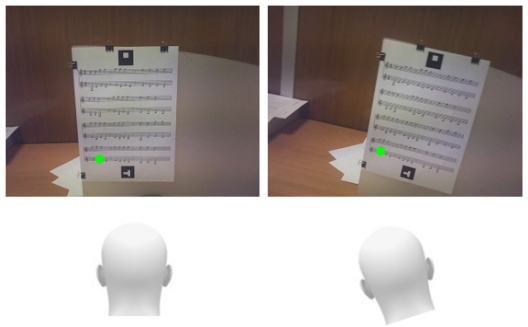


Fig. 7. Example of gaze point variation during head tilts.

### 3.6 Experimental prototype

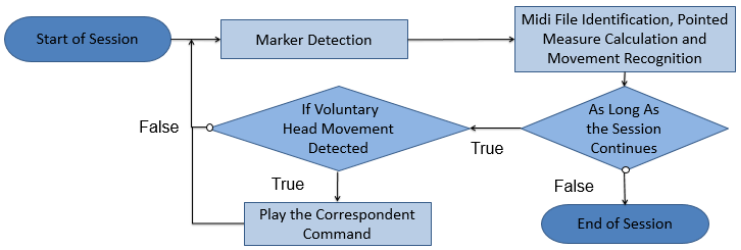



Fig. 8. Overview of EyeMusic's main algorithm.

Figure 8 shows the main algorithm of our application. The prototype we have developed for experiments is able to detect all 6 movements (M13, M31, M24, M42, M56, M65), and 6 different commands are assigned to each movement. With the help of this prototype, the user is able to play a single measure, to play from a specific measure until the end of the score, to change volume, to stop and to resume. The matching between movements and commands that we applied in our

application is shown in Figure 9. This matching was well thought out. Head tilts are used to change volume as these movements are less natural than the others and are thus assigned to secondary commands. Furthermore, they match well with the volume rotary button metaphor. Nods are used to trigger the main commands as they are easy to make. The stop and the resume are assigned to the movements which follow the same axis since they are completely opposite. In addition, we have also prepared a simplified version of our prototype, which has only one command: play the single measure that the user wants to hear. All 6 movements can be used to trigger this command. This simplified version is mainly used in the learning procedure of the experiment in order to help the participants. It is also used to evaluate the fixed-gaze head movement detection performance of our application.

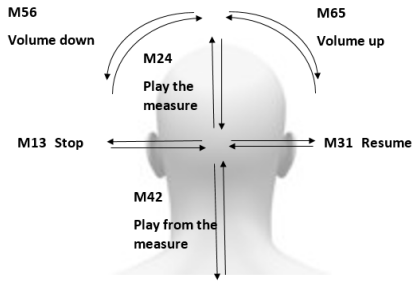



Fig. 9. Movement-command matching schematic diagram.

## 4 EXPERIMENTAL SETTING

Our experiments are divided into 2 parts. The first part is the learning procedure, during which the participants need to learn the 6 different head movements with the help of the simplified version of the prototype. The experimental data captured in this task are used to calculate and compare the success rate of each movement and thus find which movements are more suitable to be used to trigger commands. The second part is the operating task: we ask participants to manipulate the full version of our prototype to evaluate if users are capable of memorizing quickly the movement-command matching and of handling a system with multiple movements used at the same time.

### 4.1 Experimental Setup

The device we use is a head-mounted eye-tracker equipped with a scene camera produced by SMI, which we presented in section 3.1. The music score we use for tests is affixed to a board that is perpendicular to the table edge. The participants wearing the head-mounted eye-tracker sit in front of the board. They were free to move or change the distance between the board and the chair during the experiment. Figure 10 shows the setup of our experiment.

### 4.2 Procedure

The experiment takes a total of 40-90 minutes for each participant. The duration varies from person to person, because some people need more time to practice and to find the expected speed of each movement. Participants were allowed to leave during the experiment as it could take a long time.

*4.2.1 Opening.* Welcome the participants, show them a general presentation of EyeMusic written in advance, and ask them to sign the consent form. Check if the eye-tracker we use is able to

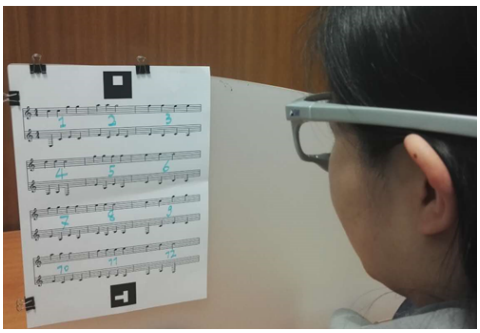


Fig. 10. Experimental setup.

capture the participant's gaze information. If the eye-tracker is not able to detect the participant's eye movements, thank the participant and discontinue the test.

*4.2.2 Learning procedure.* The simplified version of the prototype, which plays the measure of interest if any of the 6 movements is detected, is used during this procedure. We test one by one the 6 different head movements. For each movement, we present it to the participant at the beginning, giving him/her several minutes to practice and become familiar with it. Then we ask the participant to play 3 specific measures from this movement. If the music is not played successfully, we ask the participant to try again until he/she fails 5 consecutive times on the same measure. To simplify the task for participants who cannot read music, the corresponding measure number is written in the middle of each measure, as can be seen in Figure 10. The 3 measures are carefully chosen to be spread out on the score, that is to say, the 3 measures are always distributed in 3 different lines and 3 different columns. The corresponding measure numbers are presented one by one by the moderator, and the participants do not know the measure number for the next trial until the end of the current trail. There are a total of 12 (2 groups of 6) different sequences. The first group contains 1, 5, 9; 3, 8, 10; 2, 7, 12; 6, 11, 1; 7, 5, 3 and 11, 9, 4. The second group includes 12, 4, 2; 8, 10, 3; 2, 4, 12; 1, 6, 8; 9, 5, 10 and 11, 7, 6. The 2 groups are used alternatively, and the 6 sequences in each group are assigned to the 6 different movements in rotation in order to maintain a balanced measure composition for each movement.

*4.2.3 Operating procedure.* The full version of the prototype, which possesses 6 different commands, is used in this procedure. We show the movement-command matching schematic diagram (Figure 9) to participants, and give them 1 minute to memorize the matching. Then we ask them to trigger several commands without consulting the schematic diagram. The series of commands we use in this test is: play from measure 3, stop, resume, play from measure 2, volume down, volume up, play measure 1, play from measure 7, stop, resume, play measure 5, play from measure 4, volume down, volume up, and play measure 6. This includes 4 times playing from a measure, 2 times stopping, 2 times resuming, 2 times turning up the volume, 2 times turning down the volume, and 3 times playing a single measure. The stop command, the resume command and the change of volume are always triggered after playing from a measure command, while the resume command is triggered when a piece of audio is stopped. As the stop, resume and change of volume are identical for all measures, they can be triggered by fixing the gaze on any measure. We ask the participants to try again if the command was not triggered successfully, until they fail 5 consecutive times for the same command.

*4.2.4 Closure.* We ask the participants to fill in a post-test questionnaire and the SUS (System Usability Scale).

### 4.3 Collected Information

The usability test comprises 3 main parts of collected information[2]. The first part consists of logs generated by our algorithm. For each processed scene image, the raw data including detected gaze point coordinates, calculated relative gaze point coordinates, derivatives of gaze point coordinates, calculated rotation angles, and so on, are kept in a log file. Thus, in the event of a non-detected movement, we can figure out which requirement in our algorithm was not satisfied by analyzing these traces later. In addition, the original scene images are also kept in order to verify if the captured gaze point is fixed on the desired measure during the head movements.

We also take some records during the test with the help of a log paper. For the learning procedure, we record the number of the measure which was actually played for each fixed-gaze head movement performed. If there was no measure played after a voluntary movement, we wrote an 0. In that case, we are able to count the number of times used for each trial. For each user, we collected 18 (3 measures *6 movements) metrics. For the operating procedure, we record the movement performed, the movement detected, and the measure number (only for the 'play the measure' command and the 'play from the measure' command, since for the 4 other commands presented in section 3.6 the measure number does not matter). Thus, we can find out if the participants were able to memorize the corresponding head movements and if the performed movements were correctly detected. For each user, we collected at most 15 metrics.

The third part consists of information collected from the post-test questionnaire and the SUS. In the post-test questionnaire, we asked the participants to answer 10 questions, including: gender, age, degree of knowledge of music notation, educational background, their fixed-gaze head movement preference, the usefulness of each of the 6 commands we applied in the prototype, the appropriateness of the movement-command matching we use, the other commands they would like to add to our application, whether this application would be useful for their music learning, and whether they are interested in participating in other tests of our application.

### 4.4 Participants

41 users (30 males/11 females), aged between 14 and 57 years old (median = 24), participated in the learning procedure of the usability test for the EyeMusic application prototype/ 34 of them (28 males/6 females) also took part in the operating procedure of the usability test. 39 of the participants (30 males/9 females) filled in the post-test questionnaire and the SUS after the tests. None of them had tried this kind of eye-tracker before. Two other volunteers were unable to participate in our experiment because the equipment we use could not track their eye movements as there were no nose rests suitable for them. One of the participants completed the experiment wearing his nearsighted glasses inside the eye-tracker, because it was too tiring for him to concentrate on the music score without his glasses. As the lenses of his glasses are narrow and without frames, the eye-tracker worked normally during his test. For the 39 participants who filled in the post-test questionnaire, we collected their self-assessment of degree of knowledge of music notation. 13 participants claim they have never acquired any knowledge of music notation, 7 participants claim to have basic knowledge of music notation, 5 participants claim to have average knowledge of music notation, 10 participants claim to have good knowledge of music notation, and 4 participants claim to have excellent knowledge of music notation. As the number of participants with 2 knowledge levels of music notation (average and excellent) is low, we decided to group them in 3 balanced

[2]Some of the collected data will be available online: https://github.com/EyetrackingHCI/EyeMusic

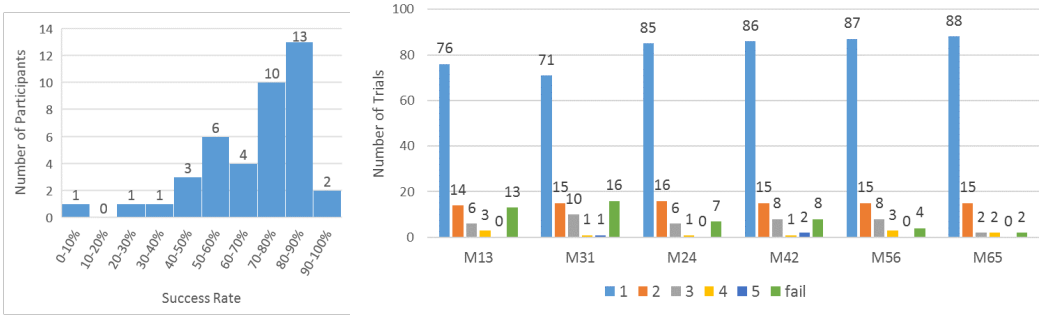
categories: 13 participants with no music notation knowledge (the target group), 12 participants with basic/average knowledge of music notation (who may also need this application), and 14 participants with good/excellent knowledge of music notation (who definitely do not need this application). When the audio of the wrong measure is played by mistake, participants with good/excellent knowledge of music notation can identify the error immediately and confidently, participants with basic/average knowledge of music notation are able to figure out the error within a short time or hesitate about the error, while participants who have never acquired any knowledge of music notation are unable to find the error by themselves. As for their educational background, 1 participant graduated from middle school, 1 participant graduated from high school, 18 participants claim to have a bachelor's degree, 17 participants claim to have a master's degree, and 2 participants claim to have a PHD.

## 5 EXPERIMENTAL RESULTS AND DISCUSSION

### 5.1 Performance of fixed-gaze head movement detection

We evaluate performance by two main measures: the average success rate, and the number of movements needed for each trial.

*5.1.1 Application performance.* During the learning procedure of the usability test with the simplified version of our prototype, we asked all participants to play 3 different measures for each fixed-gaze head movement. If the participant did not succeed, we asked him/her to try again unless he/she had already tried 5 times without success on the same measure, which meant the trial had failed. Thus, each trial can be carried out 1,2,3,4,5 times or failed. There are a total of 41*3=123 trials for each movement. As all the 6 fixed-gaze head movements correspond to the same command (play the measure) in the simplified version of our prototype, if a movement is detected as another one, we can still hear the auditory feedback. This issue occurs sometimes and affects the real success rate of our application. Therefore, we checked by hand the traces captured during the test in order to detect this kind of error. There are a total of 47 trials (out of 738) in which the feedback audio is triggered by another movement detected: more specifically, there are 8 times (out of 123) for M13, 9 times for M31, 8 times for M24, 3 times for M42, 6 times for M56, and 13 times for M65.




(a) (b)

Fig. 11. Experimental results of playing the right measure with the right movement detected: (a) success rate histogram, (b) number of trials succeeded 1, 2, 3, 4, 5 times or failed for each movement.

As we carried out these 47 trials as though they were successful, we do not know how many times we actually need to play the measure with the right movement detected. Thus, they are not counted in Figure 11b. According to the Pearson chi-squared test, there is no significant difference

between the 6 movements ($p>0.05$) when we separate them in 6 categories (trials accomplished 1 to 5 times or failed). However, if we only separate the successful trials and the failed trials, $p<0.01$ according to the Pearson chi-squared test, so the number of failed trials for M56 and M65 is clearly lower than that of the other movements.

The success rate of the application is calculated by averaging the success rate for each participant, while the individual success rate is counted by averaging his/her success rate for each fixed-gaze head movement tested. We calculated the personal success rate of each fixed-gaze head movement by dividing the number of successful trials by the number of movements performed in total. Consequently, the result does not equal the percentage of successful trials by the first movement performed. The success rate calculated is 70.0% (Figure 11a), while the standard deviation is 0.20.

Except for the average success rate and the number of movements needed for each trial, we counted and analyzed the mix-up to determine whether it is better to combine several movements to avoid the misdetection problem. The confusion matrix is shown in Table 2. In this Table we observe that a large part of misdetection occurred between head movements following the same axis, i.e. between M13 and M31, M24 and M42, or M56 and M65. These errors are easy to understand as sometimes participants make a micro movement in the opposite position before or after the desired movement in order to leave a larger amplitude for the required movement or to compensate the movement which has slightly exceeded the natural position. In addition, there are still some misdetections between M56, M65 and the other movements. This is because there may be a variation in gaze coordinates at the same time when the head tilts left or right, and participants may make horizontal head movements together with head tilts, as pure head tilts are more tiring than the other head movements. This Table shows that the percentages of misdetection for all movements are no more than 11% and thus that there is no need to combine the movements.

Table 2. Confusion Matrix.

| | Movement Detected | | | | | |
|---|---|---|---|---|---|---|
| | M13 | M31 | M24 | M42 | M56 | M65 |
| M13 | **92.5%** | 2.8% | 3.7% | 0% | 0.9% | 0% |
| M31 | 4.7% | **91.6%** | 1.9% | 1.9% | 0% | 0% |
| M24 | 0% | 0% | **93.1%** | 5.2% | 1.7% | 0% |
| M42 | 0% | 0% | 2.6% | **97.4%** | 0% | 0% |
| M56 | 1.7% | 0% | 0.8% | 0.8% | **95.0%** | 1.7% |
| M65 | 2.5% | 3.3% | 0% | 0.8% | 4.2% | **89.2%** |

*5.1.2 Movement recognition algorithm performance.* If a failure occurs when playing a measure or the command is triggered with another head movement detected, the problem can stem from 1) either the eye-tracker we used in the experiment (eye information lost, lack of accuracy), or 2) the misjudgment of users' ability (it is not easy for a few participants to fix their eyes on the measure during some of the movements or to achieve the expected movement correctly: a combination of 2 movements, an extra movement at the beginning or end) or 3) the problem of our fixed-gaze head movement detection algorithm. By analyzing the collected traces, especially the processed scene images during all the 555 failures by hand, we were able to exclude the errors caused by the other problems and concentrate on errors caused by our algorithm, that is to say, the detection failures of effective fixed-gaze head movement captured correctly by the eye-tracker. The percentage of fixed-gaze head movements correctly recognized is 84.7% with a standard deviation of 0.11 when we exclude all sources of error except those of our movement recognition algorithm (Figure 12).

Figure 13 shows that nearly all the trials were accomplished within 3 effective movements made, and according to the Pearson chi-squared test, there is no significant difference in the detection performance of the six different movements. It can also be seen that the difference in the number of failed trials discussed in section 5.1.1 is not caused by the algorithms we use for different head movements, since this no longer exists. It may be due to the difference in the degree of difficulty when making different head movements with the gaze fixed, or maybe the eye-tracker is less accurate when the user's eyes are in corners.

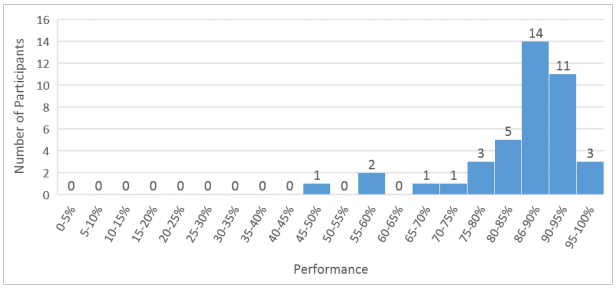



Fig. 12. Histogram of our algorithm's performance.

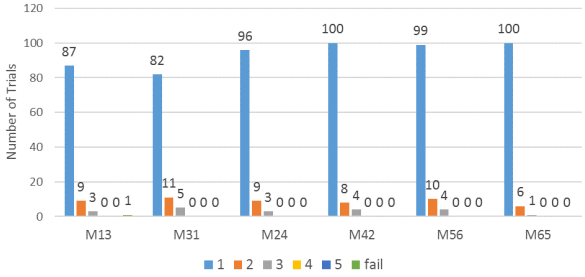



Fig. 13. Number of successful trials by 1, 2, 3, 4, 5 effective movements or trials failed for each movement.

By analyzing the origin of ineffective movements, we found that a total of 6 participants did not succeed in making effective movements that satisfied our requirement for a certain head movement tested: U11, U13, U25 and U44 failed to make M13, U11, U13, U25 and U35 failed to make M31, U26 and U44 failed to make M24, U13 failed to make M42. M56 and M65 are feasible for all participants at least once. This shows that the problems of the eye-tracker or fixation ability exist for 15% of participants, and that the great majority of participants are fine with all the head movements chosen for this application.

## 5.2 Performance of participants with different degrees of knowledge of music notation

The success rate of the learning procedure for participants with different degrees of knowledge of music notation is shown in Figure 14. It can be seen that the success rate for participants with basic/average knowledge of music notation is slightly lower than the average success rate for all the participants (70.0%), while the success rate for the other 2 categories is slightly higher than the average success rate. The distribution of the success rate is shown in Table 3. We observe that,

although the number of participants and the average success rate for each category are different, the distributions are similar: there are participants with a good success rate (higher than 80%) and participants with a poor success rate (lower than 60%) for each category. In addition, the distribution for all these 3 categories is normal according to the Lilliefors test [16].

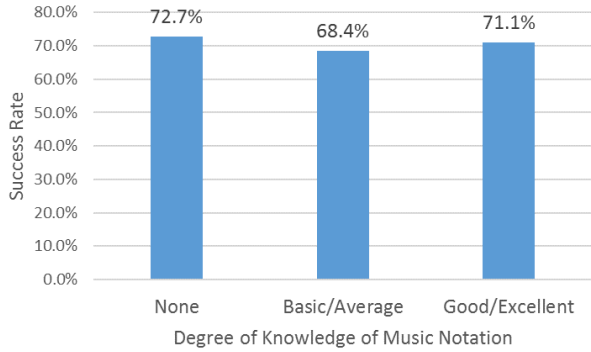



Fig. 14. Success rate of the learning procedure for participants with different degrees of knowledge of music notation.

Table 3. Success rate distribution for participants with different degrees of knowledge of music notation.

| | none | basic/average | good/excellent |
|---|---|---|---|
| 0%-10% | 0 | 0 | 1 |
| 10%-20% | 0 | 0 | 0 |
| 20%-30% | 1 | 0 | 0 |
| 30%-40% | 0 | 1 | 1 |
| 40%-50% | 0 | 1 | 0 |
| 50%-60% | 2 | 1 | 2 |
| 60%-70% | 1 | 3 | 1 |
| 70%-80% | 3 | 3 | 4 |
| 80%-90% | 6 | 3 | 3 |
| 90%-100% | 0 | 0 | 2 |
| **TOTAL** | **13** | **12** | **14** |

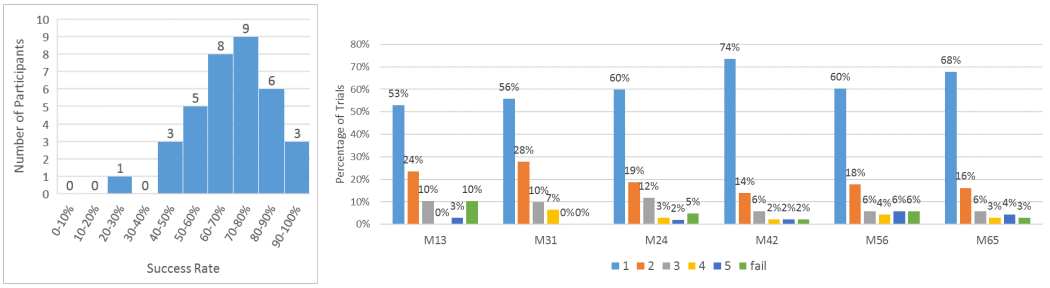
By carrying out a significance test through the analysis of variance (ANOVA), we find that there is no significant difference among the success rate of participants in different categories ($p>0.05$). Consequently, although not all the participants we recruited for this usability test are novice musicians, this does not affect the evaluated performance of our application. This is understandable because we test only the usability of our prototype, and no specific skills or knowledge levels of music notation were required.

### 5.3 Ability to memorize quickly movement-command matching and to trigger 6 different commands by these movements

For the operating procedure with the full version of our prototype, we asked each participant to trigger a total of 15 commands: 4 times to 'play from the measure', 2 times to 'stop', 2 times to

'resume', 2 times to 'turn up the volume', 2 times to 'turn down the volume', and 3 times to 'play the measure', except for 3 participants who did not perform the 'resume' command because they failed to trigger the 'stop' command first.

The average success rate is 69.3%, while the standard deviation is 0.16. This is slightly lower than the success rate of the learning procedure (presented in section 5.1) although it is far harder to use all 6 movements at the same time than to test them one by one. As the difference between these two tasks is acceptable, there is no problem for participants to trigger 6 different commands by these head movements. Figure 15 shows the results of the operating procedure of the usability test on the full version of the prototype.




(a) (b)

Fig. 15. Experimental results of triggering the right command: (a) success rate histogram, (b) number of successful trials 1, 2, 3, 4, 5 times or trials failed for each movement.

2 participants made 2 mistakes on the matching between movements and commands, 10 participants made 1 mistake, while the other 22 participants did not make any mistakes although we only gave them 1 minute to memorize it. It seems that, for all participants, it is easy to memorize the movement-command matching with a little effort. The most common mistake is the confusion between 'play from the measure' (M42) and 'play the measure' (M24).

### 5.4 Movement recognition algorithm errors and possible solutions

Despite the preliminary tests we performed with our lab colleagues, several problems arose during this usability test. For the errors caused by the algorithm during the learning procedure of the usability test, we analyzed the problem for each error and tried to figure out some possible solutions in order to improve the performance of our algorithm.

The 6 problems that can cause false negatives or false positives are as follows:

- Relative gaze coordinates are not properly calculated: there may be a significant difference between the real gaze position and the calculated gaze position, especially for the y coordinate (this occurred 39 times out of 1191 movements performed).
- Sometimes, the rotation angle is not properly calculated (this occurred 4 times).
- During head movements, the frames captured by the scene camera may not be sharp enough: thus we cannot obtain the relative gaze position as the markers are not detectable (this occurred 10 times).
- The movement speed threshold we set in the algorithm is not suitable for everybody, because participants make slower movements in the learning phase (this occurred 32 times, especially for head tilts).

- Upper body movement (forward and backward) with gaze fixed on the same point may also cause horizontal or vertical variation in gaze coordinates, and thus be detected as one of the fixed-gaze head movements we use (this occurred for 1 participant).
- When the participant is near to the music score, micro and quick eye movement which caused variations in gaze coordinates cannot be eliminated as the change of relative gaze position is too tiny and remains in the region of the same measure (this occurred for 2 participants).

The first 3 problems are mainly caused by the lack of sharpness of scene images. This can be partly resolved by using a camera with a higher frame rate that is suitable for capturing traces during movements. The 4th problem can be resolved by giving the user a longer practicing time: once the participant feels confident about making quick fixed-gaze head movements, it will be easier for him/her to interact with our application. Besides, it is also possible to set the duration threshold to an adaptive variable that can adapt to the habit of each user. The last 2 problems are linked to the distance between the user and the music score: they can be figured out by adding some thresholds connected with the distance, which can be roughly calculated from the size of the detected markers.

### 5.5 Movement preference

There is no significant difference in the detection success rate of the 6 fixed-gaze head movements, nor in the percentage of first successful trials. On the contrary, the number of failed trials for each movement is different. As shown in Figure 11b, the number of failed trials for M13 and M31 is greater than that of M24 and M42, while M56 and M65 have the least failed trials. An analysis of the number of participants who cannot make effective movements yielded a similar result.

With regard to frequency of use during the reading process, vertical fixed-gaze head movements are sometimes used to beat time (nodding). This may cause some false positives of M24 or M42, while the other movements are hardly used when reading a music score.

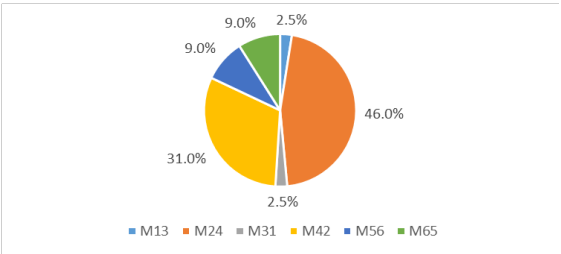



Fig. 16. Fixed-gaze head movement preference of the participants.

In the post-test questionnaire, we asked the participants to choose their favorite movement. The result is shown in Figure 16. The most popular movements are the vertical ones: M24 (selected by 46% of the participants) and M42 (selected by 31% of the participants), followed by the tilts: M56 and M65 with a support rate of 9% for each. Horizontal movements (M13 and M31) get clear low preference, maybe because people find it difficult to fix their gaze during the movement or because at times people may confuse the left and the right direction.

### 5.6 SUS result

We asked the participants to fill in the standard SUS form after the experiment. The average of the calculated score of the 39 participants who filled in this form is 77.2 with a standard deviation of

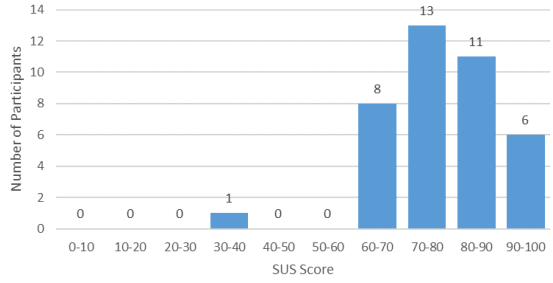



Fig. 17. SUS score histogram.

11.5. This result is good as it is higher than the average score (70) of all the tested products. The histogram is shown in Figure 17: 9 people noted our application with a score of less than 70.

We analyzed the relationship between the success rate of participants and the SUS score they assigned to our application: the result is shown in Figure 18. The radius of each bubble reflects the number of participants in each category: for example, the blue bubble in the top right corner of this figure means that there is 1 participant with a success rate between 90% and 100% who noted a score between 90 and 100 in the SUS form. The green color shows that there are more than 3 participants. It is clear that there is a strong connection between the success rate and the SUS score: participants with higher success rates are more likely to give us a high SUS score. In addition, the 6 participants who did not succeed in making effective movements for some of the head movements, scored no more than 70: this is pretty acceptable as some of the commands in this application are not usable for them.

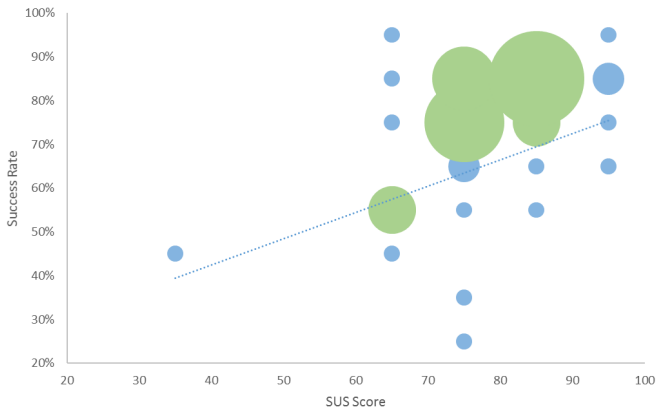



Fig. 18. Relationship between SUS score and success rate.

By comparing the score contribution for each question in the SUS form, we found that the first question ("I think that I would like to use this system frequently.") got the worst score contribution (2 on average), while the average score contribution for the other questions is higher than 2.8. This is partly because the participants we recruited are not really novice musicians: some of them already have a good knowledge of music notation so they do not need assistance in learning a melody, while some others may not be interested in music learning at all. By comparing the score

contribution given by participants with different degrees of knowledge of music notation, we find that the 13 participants with no music notation knowledge gave us the best score contribution for the first question. The average score contribution of these participants for the first question is 2.54, while 8 out of 12 participants who answered this question with a score contribution higher than 3 fall into this category. On the other hand, for participants with at least basic knowledge of music notation, only 4 (out of 26) of them answered the first question with a score contribution higher than 3, and the average score contributions for this question in these 2 categories are between 1.65 and 1.80. Table 4 shows that, although there is no significant difference in the success rate of participants in different categories, the score contribution of the first question for the group of participants with no knowledge of music notation is much higher than the other groups. Also, the average SUS score for the 13 participants with no music notation knowledge (target group) is 80.2.

Table 4. Success rate and first question's score contribution for participants with different degrees of knowledge of music notation.

| Degree of Knowledge of Music notation | Number of Participants | Success Rate | First Question's Score Contribution | SUS Score |
|---|---|---|---|---|
| none | 13 | 72.7% | 2.54 | 80.2 |
| basic/average | 12 | 68.4% | 1.67 | 75.4 |
| good/excellent | 14 | 71.1% | 1.79 | 76.1 |
| **TOTAL** | **39** | **70.8%** | **2** | **77.2** |

### 5.7 Utility of commands

In the post-test questionnaire, we asked the participants to note the usefulness of the 6 commands we applied in the EyeMusic prototype. They noted each movement by an integer from 1 to 5, and the result is shown in Figure 19. The notes for 'volume down' and 'volume up' are 3.79 and 3.77, while for the other commands the note is higher than 4.3. In their opinion, the change of volume is not as useful as the other commands as it may only be used once at the beginning. The usefulness of 'stop' is slightly greater than 'resume', which is quite reasonable as the 'resume' can only be used after the 'stop'.

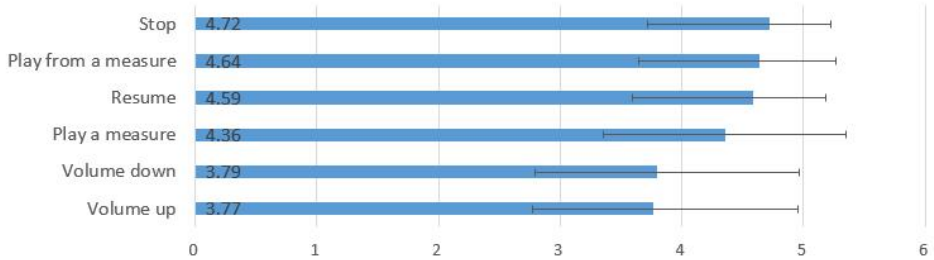



Fig. 19. Usefulness of each command scored by the participants.

We also asked them to propose some other commands they would like to add to our application. They were allowed to propose as many commands as they want, and 18 participants proposed at least 1 new command. The result is shown in Figure 20. The most popular command supported by 8 participants is to be able to speed up and slow down the music (change music tempo).

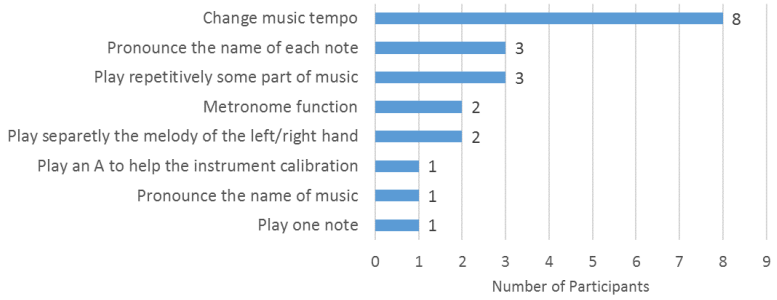



Fig. 20. Extra commands proposed by participants.

### 5.8 Utility assumption of EyeMusic

In the post-test questionnaire, we asked the participants if this application would be useful for their music learning. According to the result, 79.5% of the participants indicated that this application would be useful for the music learning process, 5.1% of the participants said they would not need it because they already possess a good knowledge of music notation, 10.3% of the participants are indifferent, and 5.1% of them do not find it useful, maybe due to the poor success rate they got during the experiment. While this question is similar to the first question in SUS, the results turned out to be different. For the first question in SUS, the average score contribution is only 2/4, but in this questionnaire, the usefulness of our application is approved by the majority of the participants. Maybe this difference is caused by the word 'frequently' in the question in SUS: although this application may be useful for their music learning, they would only use it occasionally, since it is not an application for everyday use.

Table 5. Utility assumption of EyeMusic for participants with different degrees of knowledge of music notation.

| | Degree of Knowledge of Music Notation | | |
|---|---|---|---|
| | none | basic/average | good/excellent |
| Useful | 12 | 9 | 10 |
| Indifferent | 1 | 1 | 2 |
| Not Useful (without reason) | 0 | 2 | 0 |
| Not Useful (already have a good knowledge of music notation) | 0 | 0 | 2 |
| **Total Number of Participants** | **13** | **12** | **14** |

The comparison of utility assumption among participants with different degrees of knowledge of music notation is shown in Table 5. For the 13 participants without any knowledge of music notation, 92.3% of them find it useful for the music learning process, while 7.7% of them are indifferent. The results were similar for the other 2 categories, except that 2 participants with good/excellent knowledge of music notation said they would not need this application because they already possess a good music level, and 2 participants with basic/average knowledge of music notation do not find it useful maybe due to the poor success rate.

### 5.9 Design implications

Currently, the performance of our approach when using 6 fixed-gaze head movements at the same time is fairly low compared with other interactive modalities. This problem can be solved in multiple ways. On the one hand, we can improve the performance of our head movement detection algorithm according to the solutions proposed in section 5.4, while, on the other hand, we can improve this application by modifying its design.

For example, using fewer fixed-gaze head movements would be a potential solution to make our application more reliable. This is possible in two ways. The first solution is to provide only basic commands, like to 'play a measure' and to 'stop'. The second method is to assign the commands to the head movements in a different way. For example, the 'resume' command is always used after the 'stop' command, so we can trigger these two commands with only 1 fixed-gaze head movement (if there is a piece of melody playing, this movement means to stop it, while if there is a piece of music stopped, this movement means to resume it). Furthermore, we can use the different position of gaze fixation to differentiate between commands triggered by the same head movements. For example, vertical head movements with gaze fixed on the measures mean to play the notes, while the same movements mean to change the volume (or to stop and resume) when the gaze is fixed on the volume button printed in the top left corner of the music score (or the top right corner of the paper). In this case, 2 different movements would already be enough for us to trigger all the 6 commands, and we can even add several commands proposed by the participants to our application while using fewer head movements.

The fixed-gaze head movements used to trigger commands can also be chosen in several ways. The first way is to combine movements following the same axis: in this case, we are able to eliminate the majority of misdetections, and still have 3 different movements to use. The second way is to eliminate the 2 horizontal head movements and use only head nods and/or head tilts, as head shakes are not so popular with the participants, and each one is not feasible for 10% of the users. At the same time, we can also avoid a large part of misdetection of head tilts by eliminating head shakes. Furthermore, we can allow users to personalize our application by choosing the pair of head movements they would like to use to trigger commands: users who have difficulty with horizontal and vertical head movements can use solely head tilts, while users who are fine with all these movements can choose the movements they prefer. In addition, the commands can also be personalized: users can choose the commands which are more useful for their learning phase and use their favorite movements to trigger them.

## 6 CONCLUSIONS AND FUTURE WORK

In this paper, we proposed coupling eye-tracking with fixed-gaze head movement to trigger various commands and to smooth the interaction with objects in the physical environment. We also proposed an algorithm to detect 6 different fixed-gaze head movements by the processing of scene images with gaze point overlay captured by a head-mounted eye-tracker equipped with a scene camera. To test the performance of our fixed-gaze head movement detection algorithm and the acceptance of our approach, we designed and developed an experimental application which aims at assisting novice musicians when they are learning to read music scores. With the help of fixed-gaze head movement and eye-tracking, users can continue to concentrate on the music score without disturbance or releasing their hands which are occupied by the musical instrument. Moreover, no extra device other than the head-mounted eye-tracker equipped with a scene camera is needed for this application.

We conducted extensive usability tests of the prototype with 41 people. The average success rate is 70%, although some of the participants had difficulty fixing their gaze during certain head

movements and sometimes the performance of the eye-tracker we used was not satisfactory. The performance of our fixed-gaze head movement detection algorithm is 85%, and there were no significant differences between the performance of each head movement. The average SUS score is 77.2, which is good as it is higher than the average (70), despite the fact that we got a bad score contribution for the first question ("I think that I would like to use this system frequently"). This was because some of the participants were not real novice musicians and it is not an application for everyday use. The average SUS score for participants with no knowledge of music notation is 80.2. Moreover, according to the questionnaire, 79.5% of the participants indicated that this application would be useful for their music learning process.

Despite the encouraging results, many improvements can still be made to reduce errors made by our algorithm, to make this approach more reliable, and to improve user experience. One idea is to avoid repeating the calibration process by saving the gaze parameters of each participant in a specific configuration file. We can tune the duration threshold value of each movement or set it to an adaptive variable that can adapt to each user's habit. Furthermore, we can re-design this application 1) by adding some new commands proposed by the participants to enrich the functionality of our application, 2) by using fewer fixed-gaze head movements to increase the recognition rate, 3) by changing the matching between movements and commands according to the compatibility of each movement. Subsequently, we plan to conduct a second test more focused on the utility of the application with real beginners.

For now, this application works only for music scores saved in a database. However, with a sufficiently powerful music recognition algorithm, this application will be able to work for any music score. In addition, by analyzing the user's fixation when reading music scores, it may be possible to predict whether the user has encountered difficulties and needs assistance, which would allow a pro-active behavior (commands are triggered implicitly to assist learning at the right time).

Finally, the application field of this fixed-gaze head movement detection algorithm is not limited to the EyeMusic application. By combining the technique of pattern recognition or by adding markers beside each smart object, this algorithm is able to interact with all surrounding connected physical objects, remotely with both hands free, using fixed-gaze head movements.

## ACKNOWLEDGMENTS

We would like to thank the Chinese Scholarship Council (CSC) which partially supports this research.

## REFERENCES

[1] Vasileios Athanasios Anagnostopoulos and Peter Kiefer. 2016. Towards gaze-based interaction with urban outdoor spaces. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. ACM, Heidelberg, Germany, 1706–1715.

[2] Dario Bonino, Emiliano Castellina, Fulvio Corno, A Gale, Alessandro Garbo, Kevin Purdy, and Fangmin Shi. 2009. A blueprint for integrated eye-controlled environments. *Universal Access in the Information Society* 8, 4 (2009), 311.

[3] Andreas Bulling, Daniel Roggen, and Gerhard Tröster. 2009. Wearable EOG Goggles: Eye-based Interaction in Everyday Environments. In *CHI Conference on Human Factors in Computing Systems*. ACM, Boston, USA, 3252–3264.

[4] Heiko Drewes. 2010. *Eye gaze tracking for human computer interaction*. Ph.D. Dissertation. Ludwig Maximilian University of Munich.

[5] Heiko Drewes and Albrecht Schmidt. 2007. Interacting with the computer using gaze gestures. In *IFIP Conference on Human-Computer Interaction*. Springer, Rio de Janeiro, Brazil, 475–488.

[6] Heiko Drewes and Albrecht Schmidt. 2009. The MAGIC touch: Combining MAGIC-pointing with a touch-sensitive mouse. In *IFIP Conference on Human-Computer Interaction*. Springer, Uppsala, Sweden, 415–428.

[7] Andrew T Duchowski. 2007. *Eye tracking methodology: Theory and practice*. Springer.

[8] Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2015. Orbits: Gaze interaction for smart watches using smooth pursuit eye movements. In *Proceedings of the 28th Annual ACM Symposium on User Interface*

*Software & Technology*. ACM, Charlotte, North Carolina, USA, 457–466.

[9] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. 2014. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition* 47, 6 (2014), 2280–2292.

[10] Ioannis Giannopoulos, Peter Kiefer, and Martin Raubal. 2015. GazeNav: Gaze-based pedestrian navigation. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. Copenhagen, Denmark, 337–346.

[11] Jeremy Hales, David Rozado, and Diako Mardanbegi. 2013. Interacting with objects in the environment by gaze and hand gestures. In *Proceedings of the 3rd international workshop on pervasive eye tracking and mobile eye-based interaction*. Lund, Sweden, 1–9.

[12] Anthony Hornof, Anna Cavender, and Rob Hoselton. 2004. Eyedraw: a system for drawing pictures with eye movements. In *ACM SIGACCESS Accessibility and Computing*. ACM, 86–93.

[13] Robert JK Jacob. 1990. What you look at is what you get: eye movement-based interaction techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 11–18.

[14] Takuya Kobayashi, Takumi Toyamaya, Faisal Shafait, Masakazu Iwamura, Koichi Kise, and Andreas Dengel. 2012. Recognizing words in scenes with a head-mounted eye-tracker. In *Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on*. IEEE, 333–338.

[15] Angela Kwan. [n. d.]. 6 Benefits of Music Lessons. https://www.parents.com/kids/development/intellectual/6-benefits-of-music-lessons/. ([n. d.]). Accessed March 21, 2018.

[16] Hubert W Lilliefors. 1967. On the Kolmogorov-Smirnov test for normality with mean and variance unknown. *Journal of the American statistical Association* 62, 318 (1967), 399–402.

[17] Paul P Maglio, Teenie Matlock, Christopher S Campbell, Shumin Zhai, and Barton A Smith. 2000. Gaze and speech in attentive user interfaces. In *Advances in Multimodal Interfaces ICMI 2000*. Springer, 1–7.

[18] Päivi Majaranta and Kari-Jouko Räihä. 2002. Twenty years of eye typing: systems and design issues. In *Proceedings of the 2002 symposium on Eye tracking research & applications*. ACM, 15–22.

[19] Diako Mardanbegi, Dan Witzner Hansen, and Thomas Pederson. 2012. Eye-based head gestures. In *Proceedings of the symposium on eye tracking research and applications*. ACM, 139–146.

[20] Michael Mauderer, Florian Daiber, and Antonio Krüger. 2013. Combining touch and gaze for distant selection in a tabletop setting. In *Proceedings of the Workshop on Gaze Interaction in the Post-WIMP World - ACM SIGCHI Conference on Human Factors in Computing Systems*. ACM.

[21] Apurva Mehta and Malay Bhatt. 2014. Practical issues in the field of optical music recognition. *IJARCSMS* 2 (2014), 513–518.

[22] Emilie Møllenbach, John Paulin Hansen, and Martin Lillholm. 2013. Eye movements in gaze interaction. *Journal of Eye Movement Research* 6, 2 (2013).

[23] Tomi Nukarinen, Jari Kangas, Oleg Špakov, Poika Isokoski, Deepak Akkil, Jussi Rantala, and Roope Raisamo. 2016. Evaluation of HeadTurn: An Interaction Technique Using the Gaze and Head Turns. In *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*. ACM, 43.

[24] Alice Oh, Harold Fox, Max Van Kleek, Aaron Adler, Krzysztof Gajos, Louis-Philippe Morency, and Trevor Darrell. 2002. Evaluating look-to-talk: a gaze-aware interface in a collaborative environment. In *CHI'02 Extended Abstracts on Human Factors in Computing Systems*. ACM, 650–651.

[25] Keith Rayner. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin* 124, 3 (1998), 372–421.

[26] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre RS Marcal, Carlos Guedes, and Jaime S Cardoso. 2012. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval* 1, 3 (2012), 173–190.

[27] Jacek Ruminski, Adam Bujnowski, Jerzy Wtorek, Aliaksei Andrushevich, Martin Biallas, and Rolf Kistler. 2014. Interactions with recognized objects. In *7th International Conference on Human System Interactions (HSI), 2014*. IEEE, 101–105.

[28] Jeffrey S Shell, Ted Selker, and Roel Vertegaal. 2003. Interacting with groups of computers. *Commun. ACM* 46, 3 (2003), 40–46.

[29] Fangmin Shi, Alastair G Gale, and Kevin Purdy. 2006. Direct gaze based environmental controls. In *The 2nd Conference on Communication by Gaze Interaction*. COGAIN, 36–41.

[30] Oleg Špakov and Päivi Majaranta. 2012. Enhanced gaze interaction using simple head gestures. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM, 705–710.

[31] Sophie Stellmach, Sebastian Stober, Andreas Nürnberger, and Raimund Dachselt. 2011. Designing gaze-supported multimodal interactions for the exploration of large image collections. In *Proceedings of the 1st conference on novel gaze-controlled applications*. ACM.

[32] Desney Tan and Anton Nijholt. 2010. Brain-computer interfaces and human-computer interaction. In *Brain-Computer Interfaces*. Springer, 3–19.

[33] Takumi Toyama, Thomas Kieninger, Faisal Shafait, and Andreas Dengel. 2011. Museum guide 2.0-an eye-tracking based personal assistant for museums and exhibits. In *Proc. of Int. Conf. on Re-Thinking Technology in Museums*, Vol. 1.

[34] Jayson Turner, Andreas Bulling, and Hans Gellersen. 2011. Combining gaze with manual interaction to extend physical reach. In *Proceedings of the 1st international workshop on pervasive eye tracking & mobile eye-based interaction*. ACM, 33–36.

[35] Eduardo Velloso, Jayson Turner, Jason Alexander, Andreas Bulling, and Hans Gellersen. 2015. An empirical investigation of gaze selection in mid-air gestural 3D manipulation. In *Human-Computer Interaction*. Springer, 315–330.

[36] Eduardo Velloso, Markus Wirth, Christian Weichel, Augusto Esteves, and Hans Gellersen. 2016. AmbiGaze: Direct control of ambient devices by gaze. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*. ACM, 812–817.

[37] Roel Vertegaal, Aadil Mamuji, Changuk Sohn, and Daniel Cheng. 2005. Media eyepliances: using eye tracking for remote control focus selection of appliances. In *CHI'05 Extended Abstracts on Human Factors in Computing Systems*. ACM, 1861–1864.

[38] Shumin Zhai, Carlos Morimoto, and Steven Ihde. 1999. Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, 246–253.