Personal and Ubiquitous Computing
https://doi.org/10.1007/s00779-020-01406-2

ORIGINAL PAPER

A new approach to creating and deploying audio description for live theater

Dirk Vander Wilt[1] · **Morwaread Mary Farbood**[1]

Received: 30 November 2019 / Accepted: 6 April 2020

Abstract

Audio description is an accessibility service used by blind or visually impaired individuals. Often accompanying movies, television shows, and other visual art forms, the service provides spoken descriptions of visual content, allowing people with vision loss the ability to access information that sighted people obtain visually. At live theatrical events, audio description provides spoken descriptions of scenes, characters, props, and other visual elements that those with vision loss may otherwise find inaccessible. This paper explores a novel approach to automate the creation and deployment of audio description for repeatable live theatrical events, both musical and non-musical. Using readily available tools and established sound-processing techniques, we describe a framework to automate several aspects of audio description for theater. The method uses a reference audio recording and an online time warping algorithm to align audio description with live performances, including a process for handling unexpected interruptions. We also show how a reference audio recording and descriptive tracks can be generated automatically from a show's script and used to facilitate the deployment of audio description without sufficient resources for multiple performances or a live audio describer. Finally, a software implementation that is integrated into an existing theatrical workflow is also described. This system is used in three evaluation experiments that show the method successfully aligns multiple recordings of works of musical theater and non-musical plays in order to automatically trigger pre-recorded, descriptive audio in real time.

1 Introduction

Audio description (AD) is an accommodation used by people who are visually impaired or blind. It is a spoken description of the visual elements of an accompanying work, providing an accessible alternative to obtaining information that sighted individuals at the same time and place may obtain visually. Therefore, it is temporal as well as descriptive. At live theatrical events that provide AD, such as some Broadway shows, patrons use headphones and a personal receiving device, so as not to disrupt other theatergoers. For an overview on the art and practice of AD, see Fryer [10] and Snyder [25].

✉ Dirk Vander Wilt
dirk.vanderwilt@nyu.edu

Morwaread Mary Farbood
mfarbood@nyu.edu

[1] Music and Audio Research Laboratory, Steinhardt School of Culture, Education, and Human Development, New York University, New York, NY, USA

Live theatrical events pose an interesting problem for AD services. Like fixed media, the descriptions must be timed appropriately so as not to disrupt other aural aspects of the show (musical numbers, dialogue, etc.) [10, 25]. However, since repeated live performances are by design never identical, a fixed AD track cannot be aligned in advance. In live situations, either a professional audio describer describes the visual elements in real time or a system is created to allow pre-recorded AD to be triggered [16]. Developing and deploying this type of service is expensive and time consuming. A recent study showed that media producers view AD as "a costly service with no revenue potential," [21]. According to Szarkowska [27], "A lengthy preparation process and high production costs are among the greatest obstacles to the wider availability of audio description."

This paper proposes an inexpensive and novel method to trigger AD for a live theatrical performance by only using audio obtained from a show's previous performance. The process described here warps the audio from the live show in real time to a reference recording using an established online time warping algorithm [7]. This method is a step towards being able to reduce the cost of deploying live

Published online: 08 May 2020

theatrical audio description, thus making it more available to visually impaired people.

1.1 Accessibility and live theater

Alternate formats provide important (often necessary) modes to obtain information, particularly when one mode is unavailable due to sensory impairment [22]. Researching alternate formats (including AD, sign language, and captioning) can result in additional methods and tools that may help increase overall access to information, opening worlds for audiences that may feel socially disengaged and thus less likely to participate in a particular activity. The social model of disability distinguishes impairment from disability, such that disabled individuals are not disabled by their impairments but by the barriers they face in society [19]. A common criticism of the model is that it does not provide a nuanced perspective on the "complex interplay of individual and environmental factors" [24]. In other words, it is often difficult to clearly distinguish between the impact of social barriers and impact of impairment. Despite these criticisms, the social model has helped launch the disability movement by promoting a positive disability identity and civil rights for the disabled.

AD, as a type of alternate format, attempts to remove one such social barrier. Although AD has been around for "as long as sighted people have tried to tell blind people what things look like" [20], the first scholarly work on the subject as related to vision impairment was in a 1975 thesis [9] that describes an experiment for adapting a teleplay into an all-audio format. Since then, AD quality and availability has greatly improved for different platforms, including the web [4] and fixed media [22].

Raising awareness of AD's availability and usefulness is also a concern [11]. Resources are available (such as the *Audio Description Project* [26] and the *Theatre Development Fund* [1]) to help locate audio-described live performances because in most cases, unlike audio-described film and television programs, only certain performances of a particular live show can offer AD. With active areas of research in accessible technologies, the resulting methods and tools may help address the "multiplicity of barriers and hindrances which prevent a better adoption and larger proliferation of audio description" [12].

2 Method

Live audio alignment has been successful in music score following [2] and audio-transcription alignment [15] tasks. In this implementation, a reference recording is aligned to live input using an online time warping algorithm [7]. During the live alignment, descriptive audio tracks based on their pre-aligned position in the reference recording are also aligned and played back to blind and visually impaired audience members.

First, a performance of the entire production is recorded in advance. Relevant features from that audio are then extracted and stored in frames of vectors. A second audio track is also created, containing only the AD which aligns to that recorded performance. The descriptive track is broken up into multiple smaller tracks such that one sub-track contains the audio for a single described event within the recorded performance. The points where each sub-track should be triggered are marked by frame number and stored in array F, where $F(1...x)$ represents the frame at which AD number x should be triggered. Once the marks, descriptive track, and extracted features of the reference recording are obtained, the live alignment may begin.

In the system described here, Mel-frequency cepstrum coefficients (MFCCs) are extracted from both the live input and reference recording. MFCCs are used to analyze audio for human speech and music, both of which apply in this system. The code implemented to extract MFCCs is based on [8]. In addition to 13 MFCC coefficients per frame, a first-order difference from the previous frame is appended totaling 26 features per frame. The system uses a Hamming window and 40 Mel filters. MFCC feature vectors are extracted with a frame length of 100 ms and a hop size of 40 ms. For a description of MFCC feature extraction, see [17] and [18].

2.1 Online time warping

Dynamic time warping (DTW) uses dynamic programming to recursively compare two time series U and V of lengths m and n. The output is an m by n matrix D where each element $D(x, y)$ is a cumulative (in this case Euclidean) distance between $U(x)$ and $V(y)$. The value of each cell is the cumulative "path cost" from $D(1, 1)$ up to that point [23]. Every cell in the matrix is calculated by obtaining the distance between $U(i)$ and $V(j)$, and adding it to the least of one of three adjacent (previous) cells. In this way, the cumulative path cost between $D(1, 1)$ and the current cell is determined. The smaller the cumulative cost, the better the match up to that point. When the whole matrix is calculated through $D(m, n)$, backtracking the smallest distance back to $D(1, 1)$ will be the path which relates the closest points of U to V. This algorithm requires U and V to be fully available in advance and runs in quadratic time, and so is unsuitable for a live application.

An online time warping algorithm, developed by Dixon [7], requires only one of the series to be fully available in advance, while the other series may be obtained in real time. The algorithm outputs a similar matrix D, but it builds as the input is received, one row and/or column at a time, and

only in the forward direction. Also, since it is only able to estimate the total size of the resulting matrix (the total length of the live input is unknown), it is instead bounded by a constant, which is determined in advance. Thus, it does not have the advantage of being able to backtrack from future input.

In online time warping, whenever a new frame of input is received in real time as $U(t)$, where t is both the current live input frame and the total number of input frames received so far, the system must determine whether to add another row, or column, or both, to matrix D. It does this by checking all the path cost's previous c elements of the current row t and column j of the matrix. If the lowest cost is in that row, it increments the row. If the lowest cost is in that column, it increments the column. If the lowest cost is $D(t, j)$, the current cell, it increments both. Also, if a row or column has been incremented $MaxRunCount$ times successively, it then increments the other, thus preventing the system from running away in one direction. This implementation sets $c = 500$ and $MaxRunCount = 3$ as described in [7].

Indices t and j are pointers to the current real-time position in U and V. At any point $U(t)$ (the current frame in the real-time input), the value of j is the current estimated location in V. Since index t is the current live input frame, it will always increment steadily. Index j, however, will increment based on where the online time warping algorithm estimates the current temporal location to be in the reference recording. AD is inserted based on the real-time current value of j.

2.2 Backward-forward technique

Online time warping works only in the forward direction [7], whereas classic DTW [23] makes alignment predictions based largely on future input, which is not available in a live situation. To compensate for this, Arzt [2] proposes a method for backtracking very small amounts at regular intervals during online time warping which looks into future by effectively delaying live input by only a few frames. While this method was tested in the musical theater experiments with only negligible improvement, the system implemented in non-musical works (plays) provided greatly increased mark accuracy. This may be due to the extreme variations in speaking rate between performances that was not present when a musical tempo controlled the pace of the performance.

This function works complementary to online time warping as described above, and can be enabled or disabled at any time during live alignment. Enabling it will decrease the speed in which the overall system will run, since it increases the number of calculations performed at each step during the live alignment.

The backward-forward method works as follows: After every 2 frames of live input, backtrack 5 steps in matrix D by finding the lowest cost cell (decreasing t or j) for each backward step. Then, re-evaluate the matrix from that point until reaching a current border (row or column) of the matrix. At this point, the system might arrive at the same t and j as before, but one of them may be different since future information is used. If j is lower, then the system calculates j until it reaches the original point. If t is lower, then the system waits for live input until t reaches the original point. After 2 more frames, the process is repeated.

The effect of this addition to online time warping is the ability to "re-consider past decisions" [2]. Since the backtracking brings the input 5 steps into the "past" using information that was not available when the original path was calculated, it is able to adjust itself and either find a new least-cost path for those steps, or confirm that the previous path was in fact the least cost based on the future input that had since been calculated.

2.3 Interruptions

In addition to adapting to a performance in real time, automatic AD must also account for unexpected interruptions. If an intermission, audience applause, or other such break is not included in the reference recording (which will likely be the case if a reference recording of a show's rehearsal is used), then the marks will misfire until sufficient time has passed for the marks to realign successfully once again. The longer the unexpected interruption, the longer it will take for the marks to realign. Thus, a mechanism is needed to handle these situations accordingly.

To identify interruptions, the system proposed here is trained to listen for specific audio identifiers that are representative of typical interruptions. In addition to calculating t (live input frame) and j (position in reference) with online time warping, incoming frames of live input will also be checked against these interruption frames. If the live input is closer to the interruption frames, then the system will stop incrementing j. Once stopped, the system will check against the next frames of the reference recording ($j+1$, $j+2$, etc.) with the live input. If it determines that the live input more closely matches the reference, then j will increment and the alignment will continue. During pauses, t will continue to increment since live input is still being received.

Training data is a short audio clip of representative audio that occurs during interruptions (room ambience, audience chatter, etc.). It is obtained in a manner identical to obtaining the reference recording and live input (same sample rate, hardware setup, etc.). The length of the training data need not be very long. In this implementation, multiple audio clips are obtained to represent audio from typical

theatrical interruptions, such as audience murmur during show intermissions. Each clip is then converted into a feature set in the exact manner as the reference recording, and stored in S, where $S(n)$ is the feature vector at frame n.

Detecting interruptions during live alignment works as follows: Each time the online time warping algorithm is about to increment j, a check against the interruption feature set is performed. The system finds the path cost between $U(t-c...t)$ and $S(1...1+c)$. In this way, the system determines the cost of aligning the previous c frames of live input with c frames of the interruption feature set. Then, the system finds the path cost between $U(t-c...t)$ and $V(j...j+c)$, which determines the cost of aligning the previous c frames of live input with the upcoming (future) c frames of the reference recording. If the cost is lower with aligning the live input to the reference, then there is likely no interruption. Conversely, if the cost is lower with aligning the live input with the interruption features, then there is likely a pause. The cost is calculated by finding the distance between two feature vectors at point c, then adding it to the distance between feature vectors at point $c+1$, etc., until the full cost has been calculated.

If it has been determined that an interruption is likely at frame t, then a count of how many consecutive frames of a likely pause is incremented by 1. If the number of consecutive "pause likely" frames increases beyond a set threshold, then the system will not increment j (meaning the live alignment is effectively paused). Otherwise, the system will increment j and the live alignment continues. Similarly, once a pause has been initiated, the system will remain paused until a threshold of consecutive "pause not likely" events has been reached. Increasing or decreasing the consecutive count threshold is a way to adjust the sensitivity of the system.

2.4 The alignment process

When the show begins, V contains the reference recording features, U is empty, $t=1$ and $j=1$, which are references to the indices of the first frames of U and V, respectively. Each time t increases (meaning the live recording has progressed by one frame), the new value of j is determined, based on the online time warping algorithm. If the algorithm determines that U is progressing faster than V at that moment, then increment t by 1 while a new frame of live input is received. If U is slower than V, then increment j by 1. If they are both moving at the same speed, then both t and j are incremented. Index j will keep incrementing (or not increment if the input is slower) until it matches t's estimated location, and a new t (live input frame) is obtained. The AD number x is triggered when $j=F(x)$. In this way, the descriptive tracks are able to align with the live performance based on the online time warping's estimation of the current index j of the reference.

Since the actual size of the matrix is unknown, an empty square matrix of 40,000-by-40,000 was created in the current implementation, which holds approximately 25 min of data on either t or j given the feature extraction parameters presented earlier. During the alignment, when one index is incremented up to the size of the matrix, the matrix is reset and the path cost returns to 0. In this manner, the alignment can run indefinitely, and the calculated path cost does not increase indefinitely. Other techniques may be used in different implementations, including smaller matrix sizes or a matrix that follows a window around the warp path and whose values are overwritten as the tracking progresses.

MFCCs for each minute of audio were extracted in less than 2 s while running on a 2.7 GHz MacBook Pro using a C/C++ implementation. Offline tests of the online algorithm were able to process one hour of alignment (including extraction and matrix calculation) in about 5 min. This process therefore runs comfortably in a real-time scenario.

2.5 Anchor frames

The system is able to correct itself and compensate for variations between the reference and live input. However, corrections result in misfires both of the first mark and a number of subsequent marks until some length of time for adequate correction has passed (as is demonstrated with the pause detection evaluation). When the reference is substantially different from the live input over a longer period of time or over multiple sequential marks, information about the extent of the difference greatly increases the accuracy of the triggers if that information is known in advance. For example, if the reference is used in a live session and one of the marks misfires by a large amount (several seconds or more), the theater technician could note in real time when the mark actually (incorrectly) fired, thus creating an anchor frame for that mark.

If the time between two adjacent AD trigger points is one minute in the reference recording, but it is known in advance that the duration of that sequence is anticipated to be closer to two minutes during the live performance, that information can be used to adjust the reference recording features by either removing frames or spreading out the existing frames to match the anticipated length of the live input. Thus, using these known anchor points, accuracy of the trigger marks is achieved without having to capture a new reference recording.

Anchor frames as a corrective measure work as follows: During live alignment with a previously captured reference recording, if a mark is found to be substantially misaligned, it is manually noted when the mark was actually triggered based on comparing the differences between the misaligned

mark and the previous mark in both the reference and the live input. F is the vector of reference frames to trigger mark x, L is the vector of frames in the live input when the trigger occurred during the live alignment. Ideally, frame $F(x)$ is accurately aligned with frame $L\{x)$, so mark x is triggered at the correct moment. When b is the number of the misfired mark in F and $a = (L(b) - L(b-1)) - (F(b) - F(b-1))$: If $a < 0$ then the mark occurred too early and frames must be added to the reference. Else if $a > 0$ then the mark occurred too late and frames must be removed. Else the mark is accurate and does not need adjustment. Then, on subsequent performances, one frame after the live input triggers mark $b - 1$:

```
FOR (i = 0; i < ABS(a); i++)
    IF (a < 0)
        INSERT COPY OF J(b) - i INTO J(b)- i
    ELSE IF (a > 0)
        DELETE J(b-1)
FOR (m = b; m < F.length(); m++)
    F(m) += a
```

The process adjusts the number of the reference frames to match the predicted distance between the current trigger, which is known in advance to be correct, and the next trigger, which is known in advance to be incorrect. After modifying the reference, the final step is to adjust the trigger marks in F from b to $F.length()$ to account for the change in the length of the reference. The adjusted reference recording can now be used on subsequent performances, and further adjusted as necessary by repeating this process.

3 Non-musical plays

Theatrical productions with music contain a metric pulse (tempo) and instruments that are subject only to minor variations between performances. In non-musical plays, however, there is potential for extreme variations in both speed and delivery of the dialogue. An automated AD delivery system must be robust enough to handle these large variations over a short period of time.

Modifications to the system described in the previous section allow for increased accuracy for non-musical plays (or during portions of musicals that do not contain music). Specifically, the sample rate is increased to 32,000 samples per second (four times the sample rate used in the musical theater evaluations), which increases the number of samples calculated per frame while maintaining a consistent 25 frames per second for the feature set. We also implement a normalization of the distances in matrix D as follows: For every calculated cell, the total is divided by the number of frames the path takes from $D(1, 1)$ to that point$(D(t, j)/(t + j))$. Finally, a backward-forward method is added to the alignment to account for very large changes in speaking rate.

These modifications increase the robustness while also increasing the computing power required for the online alignment. Using the same hardware, one hour of dialogue-only audio can be processed offline in about 8 minutes.

3.1 Synthesizing the reference recording and AD

Since the system described thus far requires a reference recording and a live audio describer, it is not sufficient for productions with a very limited run, without a performance to capture the reference, or without the resources to secure a live audio describer. In such cases, we propose a system to generate both the reference recording and AD for a non-musical play automatically, using the show's script and out-of-the-box text-to-speech (TTS) synthesis.

Screenplays and stage plays contain information necessary to convey the narrative of a scripted show, including character dialogue, scene setups, directions, and other business. Stage direction placed in between dialogue provides narrative context and story flow, and extracting these directions from a screenplay has been evaluated as a starting point for automatic AD creation [13]. Research on audio description versus audio narration [6] shows that narration, without accompanying description, can provide crucial details even when lacking visual information that may be present in the finished product. Research also suggests that AD read by a synthetic voice reduces the cost and increases the availability of AD [27]. Although the quality of AD created this way is up to the playwright (who was likely not writing with AD in mind), it can be used as a starting point for later, more thorough AD development.

The system proposed here accomplishes two tasks simultaneously: (1) it generates a TTS reference recording from the script that can later be used for a live performance of that work, and (2) it extracts stage directions and records appropriate mark numbers for the directions to automatically read back to the theatergoer at the correct time via TTS. For non-musical plays, this is a one-click solution that attempts to create and deploy audio description with as little setup as possible for the end user.

3.2 Method

Scripts are written in formats designed to easily distinguish between different elements, such as dialogue, character names, and stage direction. Although formatting varies between scripts, formatting within one script remains consistent. Therefore, the system can determine which written text should be used as part of the reference recording, which text can be used for the AD, and which text can be ignored, by examining the formatting and extracting

text from page layout cues. For example, in some scripts, stage direction may appear in all capital letters. In others, stage direction may be centered on the page or may be enclosed within parenthesis or brackets.

To create a reference recording and AD automatically, the system reads words from the script sequentially into a parser. The algorithm is as follows: If the word will be heard by the audience during the performance (such as a part of dialogue), concatenate that word onto the end of char array S, and continue onto the next word of the script. If the word is the first word of a stage direction: (1) send array S to a TTS converter, which returns a vector of audio samples T; (2) append T to vector V, which is the complete reference recording so far; (3) append $V.length()$, the sample number where this stage direction begins, to F, the marks of the reference recording; and (4) clear S and skip all subsequent words of the script until the parser once again reaches a word that will be heard by the audience. Repeat this process until the end of the script.

Since the reference recording was created using a fundamentally different recording process than the live input, it is necessary to standardize the dynamic range of both signals by passing both the live input and the reference recording through an audio compressor before converting the signal to feature vectors. When this process is complete, reference recording V will be ready to process for online time warping, and F will contain the mark numbers needed to trigger AD x when $j = F(x)$.

To create synthetic AD from the stage directions, instead of discarding these words when the parser comes to them, the words are sent to a TTS and the returned samples are stored as a separate AD file, labeled so that the system can identify which mark in F is associated with the direction in question. The TTS function is any out-of-the-box TTS converter where the input is an array of characters and the return is an array of samples representing the audio of the spoken text. Section 4.2 provides more details on the TTS and audio compression techniques.

4 Evaluation

To evaluate this method using a real performance as the reference, two different audio recordings of the same theatrical productions were used, with one recording as a reference and the other as live input. Markers were placed manually in both recordings to represent specific moments in the production (such as lines of dialogue or musical cues). The algorithm was then run in real time and the mark locations found during the alignment were compared with the actual mark locations in the live input.

In the first evaluation experiment, two recordings of Gilbert and Sullivan's *H.M.S. Pinafore* were used: a D'Oyly Carte recording from 1949 and a Malcolm Sargent recording from 1958. In both recordings 213 specific points were marked; these points were meant to simulate where AD may be triggered. This experiment used the D'Oyly Carte version as the reference and the Malcolm Sargent version as the live input.

After completing the alignment, the ground truth was compared with the marks automatically located by the algorithm. A total of 161 marks (76%) were found less than 1 s from the mark's actual location in the reference; 183 marks (86%) were found less than 2 s from the actual location; and 200 marks (94%) were less than 5 s. The mean difference between the marks indicated by the algorithm and the ground truth was 1.2 s (SD = 2.68 s).

To test the algorithm in a more realistic situation, a second experiment using two recordings with notable, audible differences were obtained: the Broadway (2007) and London (2010) cast recordings of *Legally Blonde, The Musical*. The London version was recorded in front of a live audience and contains audience noise, laughter, ambience, etc. that is not present in the Broadway recording. The only alteration made to the recordings was the removal of one track from the London version because it was out of order in the Broadway recording.

In both versions of *Legally Blonde*, 169 locations were manually marked, and the alignment was run in real time using the London recording as the reference, and the Broadway recording as the live input. The results showed that 117 marks (69%) were found within 1 s of the reference, 133 (79%) were within 2 s, and 147 (87%) were within 5 s. The mean difference between the found marks and the ground truth was 1.79 s (SD = 3.32 s) (Table 1).

In both experiments, the total duration of each recording was over an hour, and the algorithm was able to keep up with the long live input, and automatically correct itself after moments of difference between the reference and the live input. If there is a "mistake" between productions and the AD becomes misaligned, the algorithm may correct itself as the production progresses. For example, the longest difference between reference and live for all experiments was about 21 s, which occurred during a significant period of divergence between the two recordings of *Legally Blonde*. However, the algorithm was back to correctly

Table 1 Results of the two evaluation experiments

	Marks	< 1 s	< 2 s	< 5 s	St. Dev
Pinafore	213	75.59%	85.92%	93.90%	2.68 s
Blonde	169	69.23%	78.70%	86.98%	3.32 s

Marks refers to the total number of annotated marks for each show; values in the < 1-, < 2-, and < 5-s columns indicate the percentage of marks found within 1, 2, and 5 s of the ground truth; St. Dev refers to standard deviation of the differences between the found marks and the ground truth

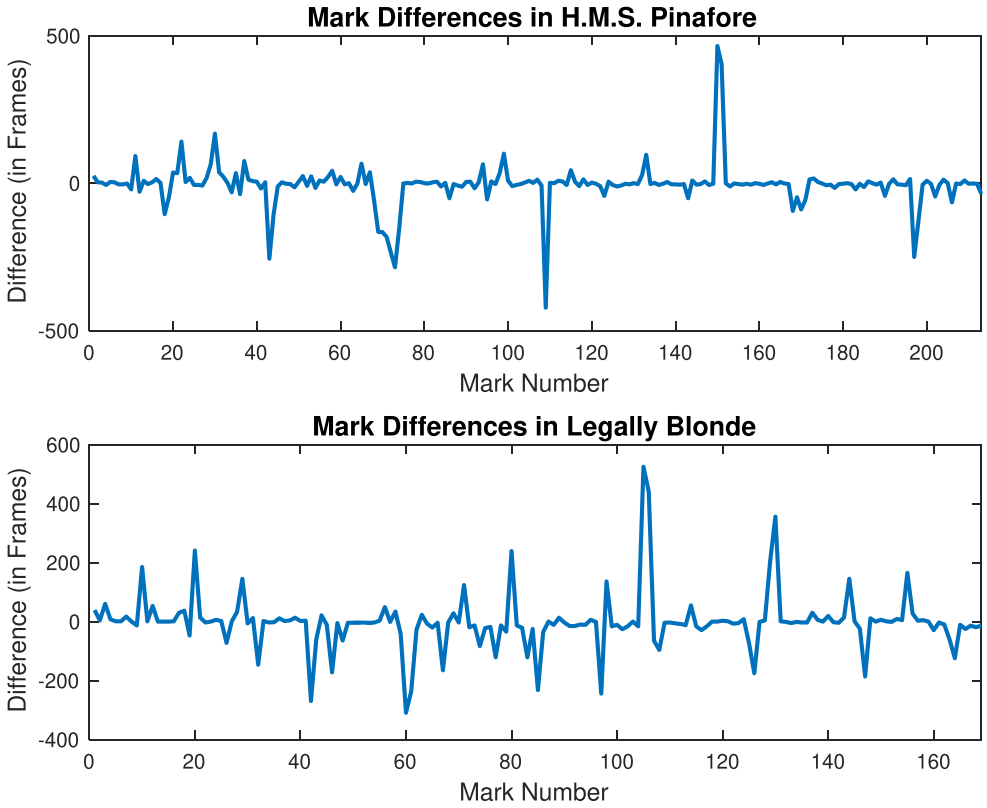

Fig. 1 Accuracy of all marks shown for *H.M.S. Pinafore* and *Legally Blonde*, shown as deviations from ground truth in frames. *X* axis indicates mark number; *Y* axis indicates difference in number of frames (25 frames = 1 s)

aligning once again less than 2 min later, with the next marks falling 120 ms from the ground truth (Fig. 1).

Within the context of a live theatrical environment, these results show that most (79–86%) AD will be triggered within 2 s of the actual event occurring. These metrics indicate that theatergoers would be able to follow the visual elements of the production in a timely way.

4.1 Evaluating pause detection

To evaluate the process of pausing the algorithm when an interruption is detected, two 3-min audio clips were obtained to represent two theatrical environments that could indicate a cue to pause. The first recording was of a small, empty theater. The second recording was of a large library atrium to represent a Broadway theater, including quiet audience chatter and activity, such as what would occur during an intermission. The experiment was performed twice, once for each audio clip. The first 30 s of each recording was used as training data, and the remaining 2.5 min was inserted after the overture of *H.M.S. Pinafore* (approximately 5 min into the recording) in the live input only. This emulates an interruption in the live performance that is not present in the reference recording, and thus makes the reference and live input substantially misaligned.

The algorithm used a threshold of 10 frames (0.4 s of consecutive input) to determine if there will be a pause. The number of frames c of which to check the cumulative path cost at each frame was set to 50. In this implementation, a random 50-frame subsection of the interruption training data was used at each frame, so as not to bias a particular part of the 30 s of training data. In these experiments, when determining the lowest cost, a weight of 2 was applied to the calculation of the pause training data only. Instead of using $if(PausePath < LiveInputPath)$ to determine if a pause is likely, $if(PausePath * 2 < LiveInputPath)$ was used. This way, the algorithm favors the alignment continuing unless there is strong indication that an interruption is occurring. In a real-world situation, the training data would be obtained at the venue prior to the performance. The weight applied, if any, would be determined by how sensitive to interruptions the show technicians would like the program to be.

First, the results of a live alignment without the interruption checking is obtained, using the modified audio file as the live input. Then, the same alignment was run again, this time with the interruption checking enabled. The results show that the mark accuracy after the interruption greatly increased when the algorithm was paused. Although both tests show that the system is able to correct itself and realign eventually, adding a check for interruptions allows the audio description to remain consistently aligned, with no period of misfires beyond what was noted in the previous evaluation. Conversely, without adding this check,

the system required about 2.3 min before it was correctly firing marks once again, which is about the same length as the interruption itself. For the first 50 marks (approximately 20 min), the average accuracy with interruption detection enabled was 1.32 s, whereas without the detection the accuracy was 16.32 s (Fig. 2).

4.2 Evaluating the synthetic reference

To evaluate the viability of using a synthetic TTS reference recording for non-musical plays, a TTS recording of William Shakespeare's *Hamlet* was generated. The script was obtained from Project Gutenberg and prepared as described in the previous section. The TTS used was Apple's macOS "Tom" system voice, accessed via the terminal "say" command. The speed of the speech playback was set to "Normal."

For the live input, a fully dramatized audio presentation of *Hamlet* was obtained from LibriVox. In both recordings, the stage directions were marked as the insertion points for the AD (in the reference, they were marked automatically; the stage directions were easily parsed in the script because they were all contained within *center* HTML tags or enclosed in brackets). Prior to extracting features, an audio compression algorithm was applied to both the reference and the live input, using the Audacity audio editing software with a threshold set to -1 dB.

For this experiment, anchor frames were also used during moments of large difference between the live input and the reference recording. Since the synthesized TTS was not designed to act in character, the lines were read in an expectedly monotonous fashion. Conversely, the live input contained sections of heavy dramatic acting. Thus, anchor frames were inserted whenever the live input deviated more than 10 s (250 frames) from the expected mark location in the reference (Table 2).

A total of 147 locations were marked in the *Hamlet* script. The same locations were manually marked in the live recording of the play. The results show that the marks are less accurate (in terms of percentage of marks under 2 s) than the musical theater experiment, but the standard deviation was lower. Therefore, this method shows promise in being able to generate a reference audio recording automatically.

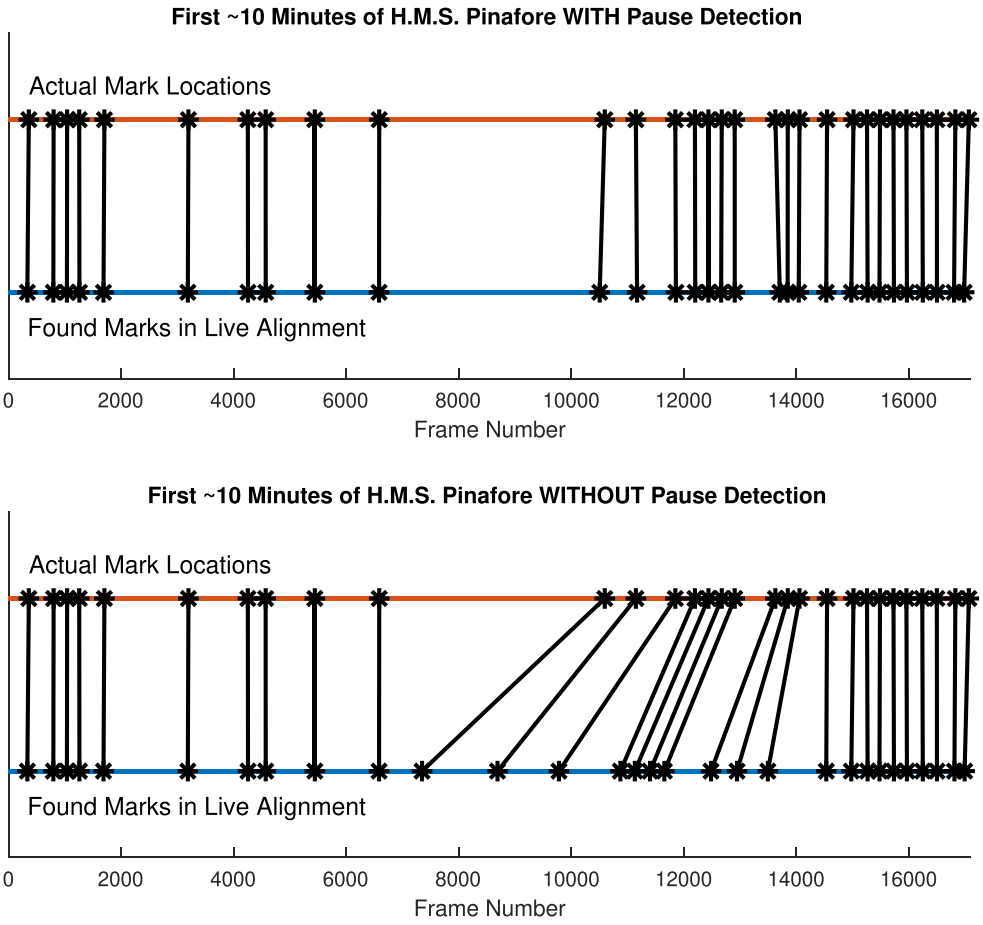

Fig. 2 Accuracy of alignment in *H.M.S. Pinafore*'s first 10 min with pause detection (top) and without pause detection (bottom)

Table 2 Accuracy of AD marks in Shakespeare's *Hamlet*, using TTS to generate the reference recording

	Marks	< 1 s	< 2 s	< 5 s	St. Dev
Act I	44	40.6%	68.6%	90.6%	2.5 s
Act II	32	50.0%	77.0%	96.2%	1.9 s
Act III	29	51.7%	69.0%	89.7%	1.7 s
Acts IV–V	42	50.0%	71.4%	90.5%	2.4 s

5 Software implementation

Software implementations to increase the availability of audio description generally take the form of automating some task of AD creation or deployment, thus decreasing cost and complexity, and ultimately increasing availability. For example, the *CineAD* system [5] uses information from existing closed captions and a teleplay or screenplay of a fixed film or television program to generate a descriptive script automatically. Alternately, *LiveDescribe* [3] and *YouDescribe* [14] assist amateur describers with creating AD for videos, including web videos.

For a real-world setting, we propose both a software implementation and workflow that takes into account existing theatrical technology (audio recording) as well as established music information retrieval techniques (feature extraction and online time warping) to align AD in real time. The previous sections of this paper have discussed the algorithm and parameters of the software; this section describes how the software may be ideally used in a live setting.

5.1 Computer and theatrical requirements

The software in this implementation runs comfortably on a 2012-era MacBook Pro with a solid-state hard drive and 16 gigabytes of memory. The computer must be able to obtain two channels of mono input simultaneously—one input to capture the reference recording and (later) the live input, and the other to capture the live audio describer. Importantly, both channels must be isolated from each other such that they do not capture audio from each other, meaning the describer must be listening with headphones so that the reference recording is not captured on the descriptive audio track. The computer must also have a single mono audio output channel to transmit the AD to audience members during the live performance.

5.2 Software interface

The software's core functions are *Record* and *Begin Show* features. *Record* activates the two mono inputs simultaneously to capture both the reference recording and the descriptive audio. The software records the reference's alignment to the AD by detecting when the speaker begins to talk, and the corresponding frame number is retained for the particular mark. Alternately, a third function, *Add Mark*, can be manually activated each time the describer wishes to add a new mark. The software may have supplemental editing features such as the ability to replace portions of the reference recording.

5.3 Capturing the Reference, Audio Description, and Interruption Training Data

With the software running, the setup configured as above, and the show commencing, the live describer activates *Record* on the interface and the system begins to capture both the performance and describer's voice on separate tracks. The system records the entirety of both mono inputs, while also capturing the sample numbers for each audio-described event. By the end, the system will have all relevant data needed to play back the description to future audiences, and a human describer is no longer needed. If interruption detection is to be used, the training data may be obtained at any time prior to the live audio alignment.

5.4 Providing AD to Audiences

A theatrical technician activates the *Begin Show* button as the production commences. At this point, the mono output (to the wireless receivers of audience members) and the mono input for the live recording (which is the same input as for the reference) is activated. The online time warping process is activated, and the AD is triggered at the correct moment.

6 Future Work

In this paper, we presented an automatic approach to triggering pre-recorded audio description during a live musical theater performance using an online time warping algorithm. The method is able to correct itself and adapt to variations between the reference and live performance, which is necessary for an effective real-time method. We also presented a method for automatically pausing the algorithm during moments of unanticipated interruptions. Although more work is needed to refine this process, it shows promise. The evaluation experiments showed that significant differences such as changes in casting, script, and instrumentation are already handled robustly.

The software implementation of the system described here can be integrated into a theater's existing setup with

minimal interference. For example, the system is able to capture, process, and deploy AD independent from the existing software or hardware controls, since it only uses an audio signal which is readily available in the performance space. No other setup or configurations are required. The simplicity of the method's technical setup and its overall flexibility provide a new way to make theater experiences for visually impaired audience members more inclusive and accessible.

Pause detection is important for theatrical works, especially given the need to respond to unanticipated audience interruptions. Longer online alignments, particularly those with multiple parts (such as breaks of indeterminate length between set pieces), need a system to respond to those events. In a truly automated, hands-off system for audio description deployment, the system needs to be robust enough to handle a wide range of interruptions that are common in theater and other long-duration live works. Further work is needed, for example, with testing additional features and different methods for detecting cues for pauses, including exploring techniques that do not require training data.

Given the proliferation of accessibility on personal computing devices and smartphones [28], using a smartphone to align and deliver the description would improve the overall success of the system. Being able to track a live performance from a mobile device without having to be connected to a wireless transmission mechanism would allow AD to be completely in control of the user, not reliant on the setup of the theater.

Audio description is a relatively new but quickly expanding accommodation for those with visual impairments. While it is becoming more common in film and television, AD for live theatrical performances remains rare. Decreasing the cost and complexity of creating and deploying AD would increase its availability, thus making the enjoyment of live theater more accessible to blind and visually impaired individuals.

Acknowledgments The authors would like to thank Juan Pablo Bello for his assistance and the Music and Audio Research Laboratory (MARL) at NYU Steinhardt.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Theatre Development Fund (TDF). https://www.tdf.org. Last accessed 24 March 2020
2. Arzt A, Widmer G, Dixon S (2008) Automatic page turning for musicians via real-time machine listening. In: Ghallab M. (ed) Frontiers in artificial intelligence and applications
3. Branje C, Fels DI (2012) LiveDescribe: can amateur describers create high-quality audio Description? Journal of Visual Impairment and Blindness 106(3)
4. Caldwell B, Cooper M, Reid LG, Vanderheiden G (2008) Web content accessibility guidelines (wcag) 2.0 WWW Consortium (W3C)
5. Campos VP, de Araújo TMU, de Souza Filho GL, Gonçalves LMG (2018) CineAD: a system for automated audio description script generation for the visually impaired. Universal Access in the Information Society. https://doi.org/10.1007/s10209-018-0634-4
6. Caro MR (2016) Testing audio narration: the emotional impact of language in audio description. Perspectives 24(4):606–634. https://doi.org/10.1080/0907676x.2015.1120760
7. Dixon S (2005) Live tracking of musical performances using on-line time warping. In: Proceedings of the 8th International Conference on Digital Audio Effects, pp 92–97
8. Dubagunta SP (2016) A simple MFCC extractor using C++ STL and C++11. https://github.com/dspavankumar/compute-mfcc. Last accessed 24 March 2020
9. Frazier G (1975) The autobiography of Miss Jane Pittman: an all-audio adaptation of the teleplay for the blind and visually handicapped. San Francisco State University, Master's thesis
10. Fryer L (2016) An introduction to audio description: a practical guide. Routledge
11. Greening J, Rolph D (2007) Accessibility: raising awareness of audio description in the UK. In: Media for all. Brill Rodopi, pp 127–138
12. Jordan P, Oppengaard B (2019) Media accessibility policy in theory and reality: empirical outreach to audio description users in the united states. In: 52nd Hawaii International Conference on System Sciences
13. Lakritz J, Salway A (2006) The semi-automatic generation of audio description from screenplays. Dept. of Computing Technical Report CS-06-05 University of Surrey
14. Lee SB (2017) Audio description in the digital age: amateur describers, web technology and beyond. The Journal of Translation Studies, 13–34
15. Lertwongkhanakool N, Kertkeidkachorn N, Punyabukkana P, Suchato A (2015) An automatic real-time synchronization of live speech with its transcription approach. Eng J-Thailand 19(5):81–99. https://doi.org/10.4186/ej.2015.19.5.81
16. Litsyn E, Pipko H (2019) System and method for distribution and synchronized presentation of content
17. Logan B (2000) Mel frequency cepstral coefficients for music modeling. In: Proc. of the International Symposium on Music Information Retrieval. Plymouth
18. Muda L, Begam M, Elamvazuthi I (2010) Voice recognition algorithms using mel frequency cepstral coeffient (MFCC) and dynamic time warping (DTW) techniques. J Comput 2(3):138–143
19. Oliver M (2013) The social model of disability: thirty years on. Disab Soc 28(7):1024–1026. https://doi.org/10.1080/09687599.2013.818773
20. Pfanstiehl M, Pfanstiehl C (1985) The play's the thing-audio description in the theatre. Br J Vis Impair 3(3):91–92
21. Plaza M (2017) Cost-effectiveness of audio description production process: comparative analysis of outsourcing and 'in-house' methods. Int J Prod Res 55(12):3480–3496. https://doi.org/10.1080/00207543.2017.1282182

22. Romero-Fresco P (2019) Accessible filmmaking: integrating translation and accessibility into the filmmaking process. Routledge
23. Sakoe H, Chiba S (1978) Dynamic-programming algorithm optimization for spoken word recognition. IEEE Trans Acous Speech Signal Process 26(1):43–49. https://doi.org/10.1109/Tassp.1978.1163055
24. Shakespeare T (2013) The social model of disability. The Disability Studies Reader, 214–221
25. Snyder J (2014) The visual made verbal: a comprehensive training manual and guide to the history and applications of audio description. American Council of the Blind Inc
26. Snyder J, Brack F (eds.) The Audio Description Project. American Council of the Blind. http://www.acb.org/adp Last accessed 24 March 2020
27. Szarkowska A (2011) Text-to-speech audio description: towards wider availability of AD. J Specialised Transl 15:142–162
28. Walczak A (2017) Audio description on smartphones: making cinema accessible for visually impaired audiences. Univ Access Inf Soc 17(4):833–840. https://doi.org/10.1007/s10209-017-0568-2