A computer implemented method of transitioning, also known as cross-fading, from a first audio track to a second audio track. A processor controls a first media player to play 101. The first audio track and output the first audio track from an audio output device. The processor also controls the first media player, or a second separate media player, to simultaneously play 103 or process the second audio track but the second audio track is not output. The processor then determines whether the first audio track and the second audio track are synchronised 104 and, if they are not synchronised the processor implements a synchronisation procedure 106 comprising pausing or halting the second track for a period of time; checking whether the first audio track and second audio track are synchronised; and repeating these two steps until the first and second audio track are synchronised. Once synchronisation is achieved, the processor controls the first and/or the second media player to transition from outputting the first audio track from the audio output device to outputting the second audio track from the audio output device. The method also includes identifying beat locations in the tracks and selecting tracks of similar tempo 102.

FIG 1
1. Play first audio track
2. Determine compatible subset of audio tracks having tempos within predetermined range
3. Receive selection of second audio track and start playing second audio track
4. Are first audio track and second audio track synchronised?
   Yes: Transition from first audio track to second audio track
   No: Adjust second audio track

FIG 1
FIG 7
The following terms are registered trademarks and should be read as such wherever they occur in this document:

YouTube
Adobe
Java
Javascript
Firefox
Internet Explorer
RealAudio
AUTOMATIC AUDIO MIXING

Field
The present invention relates to a method and accompanying system, including a user device and a server, for allowing the automatic transition from a first audio track to a second audio track, and particularly for audio tracks received from a server over a network such as the internet.

Background of the Invention
When playing a sequence of audio tracks, particularly music tracks, it is often desirable to transition from the currently playing track to the next track in an aurally pleasing way. Such techniques are sometimes known as “mixing” or “blending” tracks together. “Blending” occurs when a DJ transitions from a first song to a second song. To make the process aurally pleasing, they may gradually "fade in" the new track, while "fading out" the old one, increasing the volume of the new track while decreasing the volume of the current track. This is an example of a "blend": two tracks that are played sequentially, with a transition between the two. Although blends are often achieved manually by DJs, there have been a number of attempts to perform the task automatically, on a computer system.

The problem with such automatic computer implemented methods is that they are not suitable for implementation in a “live” scenario, particularly when the tracks in question are being streamed from a server over a network. The problem with such automatic computer implemented methods is the latency and delay involved in synchronising the two tracks. Even if the calculations are pre-prepared, it is difficult to synchronise the tracks live. In particular, the buffering required when synchronising can cause problems, as well as the fact that the latency makes it difficult to synchronise tracks by playing them simultaneously. Further problems arise because they often require complex calculations to be performed, and as such are even less suitable for implementation in a “live” scenario.

By “live” it is meant that a first track transitions to the next track without having to stop playing the first track, and whereby the second track might be selected while the first track is playing. In particular, transitioning between a first streaming track and a second streaming track may be an issue for a number of the known systems, where “streaming” refers to presenting the track to the end-user whilst the track data is being delivered by the provider, such as a server.
It is an aim of embodiments of the invention to allow audio tracks that are being streamed over a network, such as the internet, to be blended or mixed together in real time. This allows two tracks to be blended without having to stop playing the first track to allow the necessary calculations to be performed.

Summary of the Invention

The invention is defined in the claims, to which reference is now directed. Preferred features are set out in the dependent claims.

Embodiments of the invention allow users to create "blends" of any audio media from one song, or track, smoothly into the next, without technical skill or expertise. In particular, users can blend any sort of audio file, such as MP3 audio files, including audio being streamed over a computer network such as the internet, and also including audio accompanying streamed video content such as audio accompanying YouTube videos or similar.

Embodiments of the invention provide a computer implemented method of transitioning from a first audio track to a second audio track. The method is implemented using a processor of a user device executing suitable software. The processor controls a first media player to play the first audio track at the user device and output the first audio track from an audio output device associated with the user device. The processor also controls the first media player, or a second separate media player, to simultaneously play or process the second audio track at the user device without outputting the second audio track from the audio output device, for example by muting the output. The processor then determines whether the first audio track and the second audio track are synchronised and, if they are not synchronised the processor implements a synchronisation procedure comprising controlling the first or second media player to adjust the second audio track to alter the position of the second audio track relative to the first audio track by pausing or halting the second track for a period of time; checking whether the first audio track and second audio track are synchronised; and repeating these two steps until the first and second audio track are synchronised. Once synchronisation is achieved, the processor controls the first and/or the second media player to transition from outputting the first audio track from the audio output device to outputting the second audio track from the audio output device.
Repeatedly pausing the second track relative to the first to synchronise allows the system to account for imperfections of live media streaming. Unexpectedly, this iterative method works well to synchronise tracks together whilst dealing with difficulties encountered due to streaming the content from a server.

Embodiments of the invention may provide a corresponding user device comprising a processor, the user device further comprising, or being coupled to, an audio output device, the user device further comprising a communication interface for communicating with a server over a network. The processor executes a computer program that causes the processor to receive a first audio track from the server and control a first media player, executing on the user device, to play the first audio track and output the first audio track from the audio output device. The computer program further causes the processor to receive a second audio track from the server and to control the first media player or a second media player, executing on the user device, to simultaneously play the second audio track without outputting the second audio track from the audio output device. The computer program further causes the processor to determine whether the first audio track and the second audio track are synchronised and, if they are not synchronised, to implement a synchronisation procedure. The synchronisation procedure comprises controlling the first or second media player to adjust the second audio track to alter the position of the second audio track relative to the first audio track by pausing or halting the second track for a period of time; checking whether the first audio track and second audio track are synchronised; and repeating these two steps until the first and second audio track are synchronised. Once synchronisation is achieved, the computer program causes processor to control the first and/or second media players to transition from outputting the first audio track from the audio output device to outputting the second audio track from the audio output device.

The computer program may be loaded onto a memory in the user device that is accessible by the processor, or it may be provided from a remote device such as the server over the internet.

There are a number of additional features that allow embodiments of the invention to improve on the state of the art which will be described below, including: "acceptable differences" - the idea of building a list of a number of beat points (e.g. 10) for the first track and comparing them to a number of beat points (e.g. 10) of the second track, and then determining that the second track will be synchronised with the first if it is a certain period
of time apart in position from the first track corresponding to matching a beat point of the first track temporally with a beat point of the second track.

5 Brief Description of the Drawings
Examples of the invention will now be described in more detail with reference to the accompanying drawings in which:

Figure 1: is a flow chart showing a method of mixing together two tracks;

Figure 2: is an example screen shot of how representations of the audio tracks may be displayed for the user;

Figure 3: shows example waveforms for a first track and a second track to be blended together;

Figure 4: shows the result of synchronisation of the second track of Figure 3 with the first track;

Figure 5: shows a similar synchronisation to Figure 4 in which the second track is started later, during the first track;

Figure 6: shows an example of the manner in which the volumes of the first and second tracks may be adjusted during transition;

Figure 7: is a schematic example of hardware that may be used to implement embodiments of the invention; and

Figure 8: is a diagram showing the circle of fifths which is used in database form for filtering certain tracks when creating blended tracks.

Detailed Description of the Preferred Embodiments
Embodiments of the invention are implemented on a computer system. This may include user devices such as a desktop computer, laptop, tablet computer, mobile phone or smartphone, PDA and other similar device capable of processing multimedia tracks, and particularly audio tracks, in the required way. Certain processing steps may occur at a
remote server connected to a user device over a network such as the internet, with the user device receiving user input and in turn providing this user input to the remote server to select and process the multimedia in accordance with the user's requests. The remote server system preferably includes a database for storing the multimedia tracks, and is coupled to the user devices via a network, the user devices having a display and an input device such that the user can view possible tracks for selection and provide input to select one or more tracks. A description of the methods employed to select and combine multimedia tracks will be described first, followed by a description of an example system on which the methods may be implemented.

Figure 1 shows a method according to an embodiment of the invention. The method is implemented on one or more computing devices, since steps may be carried out at a user device local to the user, and steps may be carried out at a remote server. The potential distributed nature of the method will be described below. The method is implemented by one or more computer programs operating on the computing devices.

The term "media player" used herein may refer to any suitable program or application for decoding audio data and providing appropriate control signals to an audio output device, such as one or more speakers, to output the audio to a user. In particular, browser based media plug-in media players may be used, such as a "flash player", which uses an "adobe flash" plug-in to play audio music tracks, or a java script media player, such as the type used by YouTube. Also, media players based on HTML5 or similar protocols can be used to achieve the same purpose using, for example, an HTML5 based media player. As well as browser based plug-ins, application, or "app", type media player programs of the sort used by smartphones, PDAs, tablet computers and the like may be used. Any type of media player program that can be executed and controlled simultaneously with one or more further instances of such a program on a device can be used. It is possible that a single media player could be provided that is capable of decoding and controlling two separate audio streams. In this case, a single media player could be used to implement embodiments of the invention, which require independent control over different audio/media streams in order to be able to output a track and control the timing of the track relative to the other track. For the avoidance of doubt, the use of such media players is envisaged with embodiments of the invention. Preferably the audio data played by the media player, or players, is received from a remote server.
Initially, a first audio track is played by a media player at step 101. As mentioned above, this may be a dedicated media player program or application, or a web browser based media player such as a YouTube media player or similar. After selection of the first track a determination is made of compatible audio tracks from a database of audio tracks that are suitable for combination with the first track, at step 102. This selection is made based upon comparing characteristics of the first track with the corresponding characteristics of the second track. In particular, the tempo of the tracks may be compared and used to identify compatible tracks. User input is received, at step 103, selecting one of the subset of compatible tracks and the selected audio track is then played, by a media player.

The second track is "played", in the sense that the media player receives the track stream and timing data, and may also process the track data for output to an audio output device, but the audio is not output for a user to hear. The audio track is played by the media player, for example a flash player might play an mp3 track, or a java script player might play a video file with accompanying audio. One common characteristic of these types of media players is that they have various control parameters that allow the media player program to control properties of the media being played. One of these properties relates to volume. Typically, therefore, the media player is accessing the media file, such as by streaming the file from a server, for output to an audio output device, but with the volume control set to zero. The second track is therefore muted such that it is not output by an audio output device.

A check is then performed, at step 104, as to whether the first and second tracks, which are both being processed by a media player, are synchronised. The check for synchronisation involves determining the amount of time between a beat of the first audio track and the closest corresponding beat occurring in the second audio track as will be described in more detail below. If the tracks are synchronised then the media players, are controlled to transition from the first audio track to the second audio track at step 105 such that the first audio track ceases to be output to the audio output device and the second audio track commences being output to the audio output device. If the tracks are not synchronised then an adjustment is made to the timing of the second audio track relative to the first audio track at step 106. The adjustment delays the position reached by the second audio track relative to the first audio track for a period of time. This is an iterative process since the period of time is not based upon a calculation of the offset of the second track relative to the first, such as differing times between the nearest beats of the first and second tracks. The period of time is selected in a manner that allows an iterative repetition
of the adjustment until synchronisation is achieved. Once synchronisation is achieved, transition from the first track to the second track can occur.

Embodiments of the invention therefore provide a computer system implemented method in which the following steps may occur:

1. A first audio track is output by the system for user consumption.

2. Based on the first audio track, a plurality of additional audio tracks are identified by the system that may be combined or blended with the first audio track based at least on the tempo of the first track.

3. The user selects one of the plurality of additional audio tracks.

4. The system then commences processing the selected track to output it for user consumption, but mutes the track or otherwise impedes the output of the track such that the user cannot hear it.

5. The system checks whether the first track and selected track are synchronised and, if not, iteratively synchronises them by repeatedly adjusting the timing of the second track, which is muted, relative to the first, which is being output to the user.

6. Once synchronisation has been achieved, the first track is transitioned to the second track for the user to hear.

Embodiments of the invention may be implemented with two media players which can be controlled to output sound or mute the output, whilst providing the track position timing information. A program, such as a browser-based script, is used to compare the timing information of both tracks against a set of values corresponding to acceptable timing differences between the two tracks that result in synchronisation. The program or script then, if necessary, adjusts the timing of the second track relative to the first by controlling the second media player to stutter the second track it in order to synchronise the two tracks.

Embodiments of the invention therefore only require the provision of timing data of both tracks simultaneously every few seconds, and to compare the two. The two media players,
which may for example be flash or java players, play the tracks, and the program, which
may be implemented as javascript, HTML5 or flash for example, then compares the time
position data from the two tracks and stutters the tracks accordingly until they are synced.

Embodiments of the invention may be used in a system in which a client device executes
the media players locally, and the media content (i.e. the tracks) are streamed over a
network such as the internet from a server device. Typically, a first media player is
launched and commences playing and outputting a track. A second media player is then
launched, or the user manually provides input to cause a second media player to

commence playing a track. Each media player provides the position of the track (e.g. 4.34
secs) being played by the media player. Preferably the positional data is provided to a
specific accuracy such as down to two decimal places (X.XX secs). The timing data may
be provided at various times as a stream of times (e.g. 4.34secs, 5.64secs, etc), with the
media players preferably providing a constant stream that updates periodically, for example

with updates being provided every 0.01 secs.

Preferably the playback timing originates from a server, the server preferably being the
same server providing the media being played by the media player. In this case, a server
sends to the media player executing on a local client device the streaming audio/video file

plus constant/regular timing or position updates for use to determine synchronisation.

When a track is added to the server database for use by client devices in accordance with
embodiments of the invention, it is analysed for beat/beat point locations and this
information is stored on the server, ready to be called on when needed. When a user
requests to play available tracks, they are streamed from the server, or servers, to the
user’s computer device, along with live position data for the track (x.xx secs), both in real
time for each streaming media track. The user’s computer device also receives the data
indicative of the timing location of beats or beat points in the tracks.

A computer program, or script, such as a javascript within a browser or an application or
"app", executing on the user device receives data from the server(s) for the streaming
media tracks. The computer program queries the servers, or a separate database server,
for the live timing data of the tracks. For example, at any given time the computer program
will receive data such as "track 1: 0.54secs" and "track 2: 23.32secs" indicative of the
location reached by the tracks being streamed by the user device. This timing data may be
stored locally, temporarily by the local script, for use in determining synchronisation
between tracks.

The script then works out the timing differences between the two tracks being streamed, and compares these to a database of "acceptable differences", which are described below. The acceptable differences may be calculated locally by the computer program once it receives the beat/beat point data for the first and second tracks. The script then either transitions to the next track or performs a synchronisation procedure based on the acceptable differences.

In practice, there will be network related lags, and these network lags are generally caused by factors closer to the client device rather than the server. Servers delivering data are generally responsive, whereas the telephone lines and client computer receiving the data are typically less so. Using embodiments of the invention, these lags do not cause a problem because the timing details regarding track positions may be obtained or determined simultaneously for both tracks, and these may be obtained from files hosted on a common server.

For example, if we know that the first and second tracks will be synchronised when the second and first track are 1.1 seconds apart (this corresponding to an "acceptable difference" as described below) and the timing information places track 1 at 4.50 seconds and track 2 at 5.60 seconds, even if a further timing update from the server is not received for a period of time, such as 3 seconds, for any reason then it is still known that the tracks are synchronised.

- Determining Compatible Tracks

By analysing an audio file representing an audio track several key properties of the audio track can be ascertained. This includes the track tempo, but may also include the musical "beat points", which correspond to the locations of certain "beats" within the track, and optionally the track key. Using at least the tempo information allows the audio tracks to be presented to the user on a visual display, such as a visual grid, arranged by tempo. When a first track is being played, e.g. when a user selects a first track to be played, tracks that are compatible in terms of tempo for blending are determined by the system and displayed to the user.
The term "tempo" is used to refer to the timing or speed of a passage of music and is typically referred to in beats per minute (BPM). BPM will be used throughout, but it is possible that other parameters or values related to the rate or speed of the audio tracks could be used. A musical beat corresponds to the main accent or rhythmic unit in a musical piece. Generally a "beat" may correspond to a particular "hit" or note on a percussion instrument such as a drum. In particular, many modern tracks feature a bass or kick drum on every beat, or every other beat, with a snare drum often on every other beat. These bass/kick drum and snare drum events are examples of easily discernable points in a track that can be associated with a beat, and are often the aspects of a track that are used by available computer programs to determine tempo. In any case, the number of beats over a given period is indicative of the tempo of a track. A "beat point" is used to refer to a location within a track that corresponds to the occurrence of a particular beat within a sequence of beats. In particular, a beat point is the time at which a beat occurs in relation to the track. Typically a modern song may have a time signature of 4/4, indicating that there are four beats in a bar. A beat point may be defined as corresponding to the first beat in each bar, and so a beat point may occur every four beats for example.

It is not important, to the implementation of embodiments of the invention, how the audio track properties are extracted from the audio tracks. For example, the information may already be present in metadata stored with each track, and accompanying the audio data for each track when they are uploaded into the system. Alternatively, the required audio track properties may be obtained by scanning the audio data such that one or more of the tempo, beat points and key are be obtained. Existing programs are configured to extract this information from an audio track; the methods for doing so are known to the skilled person and so will not be discussed further. For example, the function "Downbeat" in the program "Auftakt" can be used to automatically calculate beat points. This allows the automatic determination of the location of the first downbeat of a track, meaning the timing of the first downbeat. For a typical 4/4 song the next downbeat may occur 4, 8 or 16 (etc) beats from the preceding downbeat. Since the tempo of the track is known, it is possible to work out the length of the number of beats between downbeats to calculate the timing of the following downbeats. The downbeat locations, or timings, may then be stored in the track metadata as "beat points" for the track.

Although each beat point in a track can be identified automatically by scanning as indicated above, knowing a beat point and the tempo allows the prediction of the location, or timing, of future beat points. For example, if the tempo of a track is 60 BPM and the first beat point
occurs at 10.0 seconds, then beat points may also be located at 14.0, 18.0 and 22.0 seconds, and so on, for tracks having beat points on every 4th beat. Although there will be other intervening beats, there are reasons for using each 4th or 8th beat as a beat point, as many songs repeat in patterns of 4-8 beats (e.g. 4/4 structured songs).

Initially, the user selects a track. The user may be presented with a selection of tracks from which to select, or the user may search for the track in the database of tracks maintained on the system. The user then selects the first track that they want to play by providing appropriate user input to the system, such as by “clicking” on a representation or icon of the track. The system then commences playing the track for the user, and determines a subset of compatible additional tracks selected from the system track database, some or all of which are then presented to the user. The subset of additional tracks is determined by the system by comparing the properties of the selected first track with the corresponding properties of other tracks stored in the track database.

The tracks in the track database are filtered based upon their tempo such that only tracks within a predetermined tempo of the first track are presented for user selection. In particular, the tempo of the first track is compared with the tempos of the other tracks within the database and only tracks with a tempo of within 5 BPM, preferably within 3 BPM or more preferably within 1 BPM of the selected track may be loaded up or presented to the user. For example, if a track having a tempo of 60 BPM were selected as the first track, a subset of tracks having tempos of between 59.0 and 61.0 BPM (inclusive or exclusive of the extreme ranges) may be presented to the user for selection. For example a selection of tracks with BPMs of 59.5, 59.7, 60, 60.1 60.3, 60.5 and 61 respectively may be presented to the user for selection.

Optionally, it is possible to further refine this process by loading tracks that are also in a compatible musical key, so that tracks that are tempo compatible (e.g. within 1 BPM) and are also key compatible have precedence and are more prominently presented to the user than those that are simply tempo compatible. As with tempo, there exist algorithms for automatically analysing an audio track and determining the key of the track. Compatible keys can be determined using relationships such as the circle of fifths, for example, which is discussed below for information.

The system may present the user with a visual indication to show that certain tracks are key compatible with, as well as being within a particular tempo of, the first track. For
example, tracks may indicate 59.5 (Key), 59.7, 60, 60.1 60.3 (Key), 61 and 60.5, whereby the reference "Key" indicates key compatibility. Here it can be seen that of these tracks that are "tempo compatible" with the original track, two of them are also "key compatible". These tracks may therefore appear most prominently on the display of the subset of tracks for selection, even though they may be slightly further away from the first track in terms of tempo. Key compatibility helps to ensure that two tracks will blend from one to the other in an aurally pleasing manner, but is not essential.

Figure 2 shows an example of how representations of the audio tracks may be displayed for the user. The audio tracks 201 may be presented to the user as a grid showing a predetermined number of audio tracks selected from a database of audio tracks. Once a first track is selected the grid may be recalculated to show only tracks that are within the necessary tempo range of the selected first track. Each representation of an audio track may be associated with an indicator showing whether the track is key compatible with the currently playing track. The tracks may be arranged such that those with the closest tempo match appear closest to the currently playing track, as demonstrated in Figure 2. Alternatively, tracks that are key compatible could be given priority, in which case tracks 98 and 72 in Figure 2 would swap positions. It is also worth noting that the grid could extend past a single display screen where more than six songs are found of a compatible tempo, with the user being able to go back and forward from one page to the next, containing different tracks, by selecting an appropriate icon 202 on the page. Preferably six audio tracks will be presented in a grid form on screen at once, although any appropriate number may be used. Each time a track is played, the grid "refreshes" based on this new selection, loading appropriate tempo/key matched tracks.

The selection of tracks may be performed manually, whereby the user selects the next track to be blended with a currently playing track each time a track plays. A first track is played, and after a time the user decides to commence playing a second track selected from the grid. The user may then select the next track to "cue" it, and the system commences the process required to segue way from the first track to the second. This can be performed repeatedly as many times as the user desires. The commencement of the method to transition from the first track to the second track may be performed immediately after selecting the second track, with crossover to the next track occurring as soon as the two tracks are sufficiently synched together. The method may, alternatively, be implemented after a predetermined period of time after selecting the next track, such as 30 seconds, 15 seconds or 5 seconds, or may be performed at a predetermined period of time
from the end of the first track, such as between 10 and 60 seconds from the end, between 20 and 50 seconds from the end or around 30 seconds from the end. Alternatively, predetermined numbers of beats may be used, such as a multiple number of beats that occur up to a particular beat point, for example the transition may occur when a multiple of 8 beats is reached, or any multiple of a number of beats corresponding to the number of beats between beat points for a given track.

When track selection is performed automatically, the user may browse through the tracks and select a list of tracks and the order in which they would like the tracks to be played, or the user may be provided with a pre-prepared playlist, which preloads tracks in a playlist. In this way the system is provided with a playlist, being a list of tracks and a desired order in which they should be played. Selection of the tracks may be performed in the same manner as described above, wherein once a particular track is selected only compatible tracks are presented for the next selection such that a list of tracks is built up whereby each track can segue into the next according to the method described herein. Once a playlist has been established, the first track plays and then after a set period, in the same manner described above for the manual process, the second track starts playing. To give the second track time to buffer and load it is preferred that the second track is loaded a predetermined period, such as 2 - 5 seconds, before a beat point of the first track. For example, if the beat point of the first track falls at 26 seconds, the system may load the second track when the first track is at 24 seconds. This aids synchronisation if the second track is commenced on a beat point of the second track, which will preferably be the first beat point, as the two tracks will start, at most, two seconds out of sync with each other.

Cue points may be provided in the metadata for each track, or may be specified by the program running on the user device. Cue points define a point within a track at which the method to synchronise and cross over to the next track may commence. This allows automatic transitioning from one track to the next at the designated cue point of the first track. Preferably, the cue point for each track is located at a predetermined time within the track, or after a predetermined number of beats, such as 32 or 64 beats from the start of the track. Cue points preferably occur on a specified beat of the track. Multiple cue points may be provided throughout a given track, and the user may determine which cue points are to be used to determine what proportion of a track is heard before it transitions to the next track. Once the cue points of a track have been determined the track may be added to the database, with the cue points logged, preferably in metadata accompanying each track.
Playlists may utilise the cue points in order to transition between tracks in an automatic manner after a predetermined time or a predetermined number of beats, which may be selected by the user.

- **Synchronising**

At step 103 in Figure 1 the user has selected the second track to play, or the system has determined that a cue point has been reached to automatically transition to the next song in a list or queue of songs. At this point the second media player commences muted playback of the second song whilst the first song continues to play, and the relative positions of the two tracks are determined to work out if the second track is "synchronised" with the first track and therefore ready to be output to the user.

Figure 3 shows the waveforms for a first track 301 and a second track 302. Each waveform shows a series of beats, for each track. These beats include beat points, e.g. beats 303 and 304, which correspond to a regular number of beats, such as every fourth beat. For illustration purposes, the tracks both begin at zero seconds, but the same principles apply if the first track had previously been playing prior to the second track commencing, as is the case for embodiments of the invention. While the first track is playing, the system tracks the play position of the track as it progresses. This may preferably be measured down to tenths of a second, hundredths of a second (0.01, 0.02, 0.03 seconds etc) or to the millisecond level depending upon the accuracy permitted by the server or media player program/application running on the computer system.

Track 1 301 and track 2 302 have similar tempos, but as can be seen from the figure are out of synch, or out of phase, with one another because the beats of track 2 are not cotemporaneous with track 1, in that they do not occur at substantially the same time. The second track commences playing as described above, with playback preferably commencing from the point of the first beat point, but with no volume (note that Figure 3 does not show playback starting at the first beat point 304). Once the second track begins muted playback, the next available beat point 303 of the first track is determined from metadata accompanying the audio track or read from a memory and the time at which this beat point occurs is compared to that of the next closest beat point 304 of the second track, again using track metadata. As an example, track 1 may have a tempo of 60BPM, with the first beat point occurring at 10.0 seconds. If the beat points are defined as every fourth
beat then for track 1 these will occur at 10.0, 14.0, 18.0, 22.0 seconds and so on. If track 2 has a tempo of 60BPM also, but the first beat point occurs at 15.0 seconds, then the beat points occur at 15.0, 19.0, 23.0, 27.0 and so on.

5 The next step is to try and match any of the beat points from track 2 to that of track 1. Embodiments of the invention may use a concept of “allowable differences” between tracks. An allowable difference is a timing difference between track 1 and track 2 that corresponds to the difference in the location of a given beat point in the first track and a corresponding beat point in the second track. For example, an allowable difference may be the difference in the time location of the first beat point of the first track with the time location of the first beat point of the second track or vice versa. Since tracks are periodic in nature, an allowable difference may also include the timing difference between the $n^{th}$ beat of the first track and the $m^{th}$ beat of the second track. An allowable difference therefore corresponds to the relative timing difference between the first track and the second track that would result in the first and second track being synchronised when played simultaneously.

In particular, synchronisation may be determined by comparing the difference in time between a beat, or beat point, of the first track/video with that of the second track/video.

10 This can be implemented by determining the position of the first beat in the first track, and the first beat of the second track, and calculating the difference between these. The tracks will be synchronised when the difference in their positions, preferably measured from the start of both tracks, is equal to, or within the synchronisation limits of, the difference between first beat point of the first track and the first beat point of the second track.

15 This method can be expanded further, by including in the track metadata one or more fields for “differences”. These “differences” are acceptable differences between the position of the first track and the position of the second track. Rather than having only one timing value for the possible difference between the first and second track required for synchronisation there are a number of possible differences that would work equally well. For example, if a track has a tempo of 60 BPM, there is 1 beat every 1 second. If it was 90 BPM, then there would be a beat every 0.66 secs, and so on. The allowable differences, which are the separation between the two tracks that would still allow synchronisation to be achieved, can be calculated from the following formula:

20 Difference 0 = difference in first beat points
Difference 1 = difference in first beat points + length of beat
Difference 2 = difference in first beat points + length of 2 beats
Difference 3 = difference in first beat points + length of 3 beats
etc.

This formula can be generalised as:

Difference X = difference in n\textsuperscript{th} beat of first and second track + (length of beat multiplied by y)

Where n, and y are any positive integer.

For example, if the first track has the first beat, such as a snare drum or other transitory rhythm note at 0.40 secs, and the second track has the first beat at 3.52 secs, the difference, difference 0, is 3.12 secs. If both tracks are 60 BPM then synchronisation could occur when the difference between the positions of the two tracks is 4.12, 5.12, 6.12 seconds and so on, these values being the base difference plus the time of n beats. If the tracks were 90 BPM, sync would occur at: 3.78, 4.44, 5.12, 5.78 etc.

More specifically, the acceptable differences may be limited to differences between beat points, rather than beats. In this instance, the value of "y" in the above formula would correspond to any multiple of the number of beat periods between beat points. For a track having a beat point every 4 beats, the value of y would be any multiple of 4.

Preferably, for any given track/video, metadata is provided including the timing location of the first beat relative to the start of the track or an appropriate marker or cue point, and the tempo of the track. As mentioned above, these values can be determined automatically by appropriate algorithms analysing the audio data. Preferably the system is arranged to then calculate, using the metadata for both tracks, one or more of the abovementioned acceptable gaps and store them into the "differences" field in a database of a store accessible by the system, or in memory, such as a RAM, accessible by the system. These numbers may be generated when the video/track is uploaded into the system database, or may be generated at the time the instruction to start playing the track commences, for example. When both videos are playing, the algorithm will start tracking the difference between the location of the first video and the second video, again relative to the start of the videos or some other prearranged cue point, and compare the difference with the
calculated multiple allowable differences. If no difference is found matching the allowed
differences, or within the timing range allowed for synchronisation, then the algorithm will
synchronise the two videos in any of the manners described herein, particularly whereby
quick variations between playing, pausing and playing again will randomise the position of
the second track relative to the first and change the difference. Once the tracks are
synchronised, transition to the second video can occur in any manner described herein.

By determining the location of beats of a track in relation to the playback time of the track, it
is possible to compare the playback time of the first track with the playback time of the
second track to establish whether a beat point of the first track is sufficiently close in time to
a beat point of the second track in order to be considered synchronised, or whether the
timing between the first track and second track is sufficiently close to an allowable
difference to be considered synchronised. This could be achieved by determining the
playback time of the first track and the second track and comparing the relative positions of
the two tracks. The relative position can be determined by a direct comparison of the times
reached by both tracks. If the first track is at time \( T_1 \), and the second track is at time \( T_2 \)
there will be a beat on the first track within a time \( T_a \) of \( T_1 \) and there will be a beat on the
second track within a time \( T_b \) of \( T_2 \). If the difference between \( T_a \) and \( T_b \) is within the
margin allowed for two tracks to be considered synchronised, then the system will proceed
to the next steps outlined herein. Alternatively, the relative positions can be determined by
subtracting the position of track 2 from the position of track 1 (or vice versa) and
determining whether this corresponds to an “allowable difference”.

In the above example tracks 1 and 2 both have tempos of 60BPM, track 1 has beat points,
defined for every fourth beat, occurring at 10.0, 14.0, 18.0, 22.0 seconds and so on, and
track 2 has beat points, defined for every fourth beat, occurring at 15.0, 19.0, 23.0, 27.0
seconds and so on. For example, it would be acceptable for track 1 to be at 10.0 seconds
when track 2 is at 19.0 seconds; track 1 to be at 18.0 seconds and track 2 at 27.0 seconds;
or track 1 to be at 22.0 seconds and track 2 at 15.0 seconds and so on. The allowable
differences between the first and second tracks may be any differences in timing between
the first track and the second track that correspond to an alignment of beat points. It is
possible to simply match any beat of the first track with any beat of the second track, with
allowable differences being determined for each beat pairing between the tracks, or a
selection thereof. However, as mentioned above, the concept of beat points may be used
such that syncing occurs at a more natural location within the track, corresponding to the
start or end of a phrase or bar of music as indicated by the beat points. The common
period is 4 beats to a bar, such that a beat point occurs at the start, or end, of each bar and synchronisation occurs on this basis. However, other distributions of beat points are possible, such as every two beats, every three beats and so on. Further limitations may be placed on the locations in which the second song may synchronise with the first, such as downbeats as mentioned above.

A database of timings of beat points for each track being used in accordance with embodiments of the invention may be stored on a server, which may be the same server as stores the media tracks, or may be a different server. When the user requests a track this information is called by, and provided to, the local user device, and preferably to a program such as a browser script or app operating on the user device. The computer program uses the beat point information for the first track and second track to compile a list of acceptable differences in timings between the first and second track which are used for synchronisation determination.

Synchronisation is deemed to have occurred if two beats, or, preferably, beat points, occur within a predetermined time of one another. In particular, if the beat points are located within 0.1 seconds of one another or less then this may be deemed to be synchronisation. Generally up to a 0.1 second gap between beat points is not noticeable by the user, but a greater difference would be noticeable. Other maximum gaps between beat points may be used to determine if synchronisation has occurred, and the threshold may be reduced below 0.1 seconds. For example, a gap of between 0.00 to 0.05, or 0.00 to 0.04 seconds, or approximately within 1/20 of a second, between beat points may be classed as synchronised, with everything outside the range classed as unsynchronised.

In the event that the first and second tracks are not synchronised, the second track is adjusted by a particular amount. The manners in which the second track may be adjusted are described below. The common property of the various methods is that the system automatically adjusts the second track relative to the first by a value that is not related to, or calculated from, the detected difference in timing between tracks determined by the difference in timing between neighbouring beats or beat points of the first and second tracks. The adjustment is applied “on the fly” whilst the first track is being processed, or played, by a media player and whilst the second track is being processed, or played, by the media player or a second instance of a media player, but not output for consumption by the user.
The adjustment of the second track may be achieved by holding, or pausing, the track for a period of time, or skipping the track on by a period of time such as by speeding the track up for a period of time. To an extent the method used may depend upon the controls available within the media player software used to play the second track.

In one embodiment, the second track is paused for a predetermined period of time, such as between 100-1000 milliseconds and preferably between 100-300ms or 300-600ms. The aim is to randomise the timings between the tracks such that the next beat of the second track is moved within the time period close enough to a beat of the first track in order to be synchronised. The greater the time for which the second track is paused, the more likely a beat of the second track is to be moved into the required time period, but the longer the user needs to wait. 300-600ms is a preferred range to balance these two factors. Once the predetermined period of time has elapsed, the second track commences playback slightly delayed relative to the first track. The second track plays for a predetermined period of time, such as 1 or 2 seconds, during which a check is performed by the system, as described above, to see if the second track is now synchronised with the first. This process is repeated until the tracks are synced in an iterative process known as "stuttering".

For the avoidance of doubt, the algorithm does not need to wait until a beat point occurs in order to determine if synchronisation has occurred. The positions of each beat, or beat point, are known from, or can be calculated by, the accompanying track metadata. The algorithm can therefore monitor the relative positions of the first and second tracks and determine from this whether acceptable differences have been reached meaning that synchronisation has occurred. The algorithm can determine the relative difference at any point in time and synchronise based on the relative difference since the positions of the beat points are contained in, or can be calculated from, metadata available to the algorithm.

This iterative process will, by sheer process of attrition, eventually produce a synchronisation of the two tracks. Advantageously, and unexpectedly, in almost all cases synchronisation is found before the first track has faded out and transitioned into the second track. If synchronisation is not found by the time the first track reaches the final beat point, the system may still control the second media player to output the second track.
Alternatively, if synchronisation is not found after a predetermined number of attempts, a predetermined period of time or a predetermined number of beats, such as 16 beats, the second track may be restarted from the first beat point and the process of synchronisation begins all over again. If one or more subsequent attempts are required then the predetermined period of time used to pause the second track relative to the first track can be varied to a new value for the duration of each of the subsequent attempts. For example, the first period may be 300 milliseconds, the second period may be 200 milliseconds, and a third period may be 100 milliseconds. Alternatively, the first period may be 600 milliseconds, the second period may be 500 milliseconds, and a third period may be 400 milliseconds.

Alternative manners in which the second track may be adjusted are possible. The method above relies on using a predetermined period of time, whereby the system is configured to pause or stutter the track by a particular period of time. However, adjustments are made repeatedly in the iterative process until synchronisation occurs, and the period of time of the adjustment may be varied in other embodiments of the invention. The adjustment period may be altered every time a new adjustment is made, or after a predetermined period of time, number of beats, or after a predetermined number of adjustments have been made, such as 4 or 5 adjustments.

The alterations to the adjustment period may be made in a predetermined manner, whereby the alterations change according to a predetermined scheme. For example, the adjustment period may decrease as the number of adjustments made prior to synchronisation increases. In this manner, the adjustment period becomes smaller, or finer, the closer the second track gets to being synchronised with the first. This helps to avoid the adjustments to the second track overshooting and requiring another cycle of adjustments to achieve synchronisation. If synchronisation does not occur by the time the adjustment period reaches a predetermined lower limit then the cycle may repeat, with the adjustment period being increased back to the starting value. A predetermined number of cycles may be permitted, such as three, before the system deems that synchronisation has not been found at which point, as above, the system may cause the second track to be restarted from the first beat point and the process of synchronisation to begin all over again.

Alternatively, the initial adjustment period may be a random value generated below an upper limit, and preferably between an upper and lower limit, for example between 10 and
300 milliseconds, or between 300 and 600 milliseconds, and each subsequent adjustment period may also be a randomly generated value such that each adjustment has a different, randomly generated, period. Possible methods for producing random values between upper and lower limits are well known and will not be discussed further here.

Whatever synchronisation method is used, generally the tracks may begin playback from the start (0.0 seconds), or from a first cue point defined in the metadata accompanying the track (for example with the first snare drum hit, e.g. 1.1 seconds). Within certain systems in which the invention may be employed, such as the YouTube API, it may only be possible to control videos to be loaded up to the nearest full second, without decimal places. To accommodate this, any differences may be rounded up or down as appropriate; for example a cue point of 1.1 seconds could be rounded down to 1 second.

Figure 4 shows the result of synchronisation of the second track 302 with the first track 301. As can be seen, a beat point 303 is aligned with beat point 304 such that they occur at substantially the same time. Figure 5 is also shown to clarify, for the avoidance of doubt, that synchronisation can be achieved where the second track is started later during the first track by matching beat point 303 of track 1 with a different beat point, 305, of track 2. Provided any of the beat points match, substantially all beat points will match, since the tempos are the same.

- Transitioning

As well as synchronising the first track with the second track, it is necessary to transition from outputting the first track to the user for aural consumption to outputting the second track to the user for aural consumption. Preferably this is performed automatically by the system, and these are the embodiments that will be concentrated on below, but it is also possible that the user could control this process manually via user inputs. Whilst the transitioning process may be implemented after the first and second tracks are synchronised, the transitioning process may instead be initiated during, or at the same time, as the synchronisation process is initiated. In either case, live or “on the fly” transitioning between tracks can be achieved whereby seamless transitioning may be initiated as soon as, or soon after, the instruction to blend two tracks is received by the system.
The general process of transitioning from the first track to the second track involves controlling the media player to lower the volume of the first track ("fade out") while increasing the volume of the second track ("fade in"), by using control functions associated with the media player. Simple embodiments may achieve this by lowering the volume on the first track over the course of a predetermined period of time while increasing the volume on the second track over the same, or substantially the same, period of time, such as 8 beats of the first or second track. After a few seconds the second track will be at full volume and the first track will have no volume. The first track can then cease being played or processed by the media player and the second track continues to play.

In manual embodiments in which tracks are selected and transitioning is initiated instantly, or within a predetermined period or at a predefined point in time from the instruction to transition, compatible tracks that may be blended with the second track may be presented to the user, with the compatible tracks being arranged by tempo as described above. This process can be repeated ad infinitum.

Figure 6 shows an example of the manner in which the volumes of the first and second tracks may be adjusted during transition. Before synchronisation is found, the first audio track 601 may be at 100% volume and the second audio track 602 may be at 0% volume, for example. When synchronisation is found the volumes of both tracks will change in the manner indicated. As a further illustration, the volume changes may appear as in Table 1 below.

<table>
<thead>
<tr>
<th>TIME (seconds)</th>
<th>VOLUME - Track 1</th>
<th>VOLUME - Track 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>80</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>60</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>80</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

**TABLE 1**

The values shown in the table are illustrative only, and other values may be used. Preferably the fade in/fade out transition time is approximately roughly 5-10 seconds in duration. Although linear fades are shown, other functions or shapes of fade in or out are possible. Once the fade process is complete, the second audio track will continue to play on full volume until it finishes or a subsequent track is selected for blending.
As described above, the system allows tracks having a tempo of within a predetermined threshold of each other to be selected for the blending process. Utilising methods such as time stretching to match the tempos of audio tracks would require processing that may not be possible in real time, and that would not work desirably when the audio is being output at the same time as accompanying video (e.g. music videos). For tracks that do not share a common tempo, the synchronisation process will cause a beat or beat point of the second track to align with a corresponding beat or beat point of the first track. However, where two tracks of slightly different tempo are being blended, the subsequent beat points will drift further apart in time as the two tracks progress. Although small differences are imperceptible to the average user, the cumulative effect will quickly result in two unsynchronised tracks. The transition period may, therefore, be selected based upon the time it takes for two audio tracks to drift out of synchronisation.

Although bigger variations in tempo are possible, if a bigger difference is used, such as 5 BPM, then the two tracks will drift very quickly. For example, if one track has a tempo of 70 BPM, and the other 75 BPM, then they would have beat lengths of 0.857 seconds and 0.8 seconds respectively, meaning a difference of 0.057 secs. Even if the tracks are synchronised as described above they will gradually drift out of sync. Table 2 shows how subsequent beats of the second track drift after synchronisation.

<table>
<thead>
<tr>
<th>BEAT</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIFFERENCE</td>
<td>0.057</td>
<td>0.114</td>
<td>0.171</td>
<td>0.228</td>
<td>0.285</td>
</tr>
</tbody>
</table>

**TABLE 2**

Even after just 5 beats, the tracks are out of sync by over 1/4 of a second. This would not sound good to the average user. Tempos of 70 BPM and 72 BPM result in beat lengths of 0.857 seconds and 0.833 seconds respectively, meaning a difference of 0.024 seconds. Table 3 shows how these subsequent beats of the second track drift after synchronisation.

<table>
<thead>
<tr>
<th>BEAT</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIFFERENCE</td>
<td>0.024</td>
<td>0.048</td>
<td>0.072</td>
<td>0.096</td>
<td>0.12</td>
<td>0.144</td>
<td>0.168</td>
</tr>
</tbody>
</table>
TABLE 3

After 7 beats (approx 5 seconds), a noticeable drift of 0.168 seconds arises. Allowing a
gap of 1 BPM results in less drift, as shown in Table 4 for tempos of 70 BPM versus 71
BPM.

<table>
<thead>
<tr>
<th>BEAT</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.012</td>
<td>0.024</td>
<td>0.036</td>
<td>0.048</td>
<td>0.060</td>
<td>0.072</td>
<td>0.084</td>
</tr>
</tbody>
</table>

TABLE 4

As can be seen, after 7 beats, the gap between respective beats is only 0.084 seconds,
which is considerably less noticeable to the average user.

Clearly the difference between beats for a given difference in BPM varies inversely to the
product of the BPM of the first and second audio tracks. The slower the tempo, or the
lower the BPM value, the greater the difference between beat lengths and the quicker two
tracks will drift from synchronisation. Using 1 BPM as the acceptable variation of tempos
for blending also applies over a sufficient range of tempos and has been found experimentally to enable the method to work for most practical BPM ranges.

Taking the above into account, the quicker the transition period between tracks (the fade in
and fade out), the less drift there will be over this period for tracks with slightly differing
tempos, for example, if two tracks of tempos 70 BPM and 71 BPM respectively are fully
mixed by beat 7, the difference at this point is only 0.084 seconds, but if it takes until beat
14, then the difference will be 0.168, which is much more noticeable.

In addition, the tempo value of any audio track as contained within track metadata stored
within the system database may be converted to a number below a first threshold and,
preferably, above a second threshold. For example, the tempos may be required to be
between 50 BPM and 100 BPM. If the tempo of a track when being added to the database
exceeds the upper threshold then the tempo value is halved, for example 102 BPM is
halved to 51 BPM. If the tempo is below the second threshold then the tempo is doubled,
for example 48.5 BPM, to 97 BPM. The tempo value of a track should, preferably, be
between 50 BPM and 100 BPM and if it is not, then the value should be converted to a
number that is. To be clear, it is not the tempo of the track itself that is being edited to between the threshold values, it is the metadata value of the tempo that the system uses to perform grouping of audio tracks for blending. The BPM values in the metadata can be edited so that they are adjusted between the upper and lower thresholds.

The point at which the transition from the first track to the second track occurs can be controlled, either by the user or by the algorithm. In particular, embodiments of the invention may allow a track to be synchronised, and the transition may occur at a later point. The user may select a first track, which commences playback. While the first track is playing, the user may select any other track, which will commence playback without outputting the track to the speaker. The second track remains muted while the algorithm determines if the tracks are synchronised, and if not then the synchronisation process is implemented as described herein. The second track may then remain muted, in a synchronised state that is ready to be transitioned to when required.

The status of the track may be indicated by a visual indicator associated with the track icon. For example, in the example layout of Figure 2, a synchronised track that is ready to be transitioned to could be represented by a green button, or background behind the track icon. Receiving user input such as clicking on the track or the green button will start the transition from the first track to the second track. Preferably it is possible to cue, or synchronise multiple tracks at the same time. All the selected tracks will be processed by the algorithm, allowing the user to select whichever of the tracks they desire, once synchronisation is established.

Alternatively, the transition between tracks may be controlled by the algorithm, whereby transition is initiated a predetermined period of time, or predetermined number of beats, from synchronisation, or from the instruction to initiate blending. For example, the transition may occur 15 – 20 seconds after synchronisation. As another alternative, the entire process of synchronisation and transitioning may be implemented a predetermined period of time, or number of beats, from the initial instruction to blend two tracks.

A particular embodiment of the invention relates to use with embedded media players such as the You Tube media player. An embodiment will briefly be described which uses the YouTube API to blend between two tracks, which may have accompanying video. For the following description it is the audio component of the video that is used for processing and
calculation purposes, although the video component will be adjusted during the synchronisation period as a by-product as the position of the second track is adjusted.

The YouTube API allows YouTube videos to be embedded onto a website and perform a number of additional functions. These include controlling embedded videos by playing, pausing and skipping to a specific point, which may be rounded up to the nearest second. In addition, control over volume is provided. Furthermore, it is possible to read data about the video, such as identifying if they are playing or paused, and the position of the video, preferably in milliseconds, from the start (e.g. "5.055 secs from start").

Whilst a YouTube specific embodiment will be described, embodiments of the invention may function with any video embedding software that provides the following functionality:

i) The ability to extract the position reached by the video, preferably to an accuracy of the nearest millisecond. This allows a check of the relative positions between the two videos, and to compare the relative positions of the 2 videos' cue points, or beat points, to see whether they are synchronised or not.

ii) Basic play/pause functions to allow the "stutter" of the second video if the videos are not synchronised.

iii) The ability to skip to a specific part of a video, such as allowing "seek" to a specific point in the video. This can be used to load cue points, or to skip through a track that is not perfectly synced.

iv) The ability to change the volume of the audio accompanying the videos. This allows two videos to be played concurrently and then "fade in" and "fade out" between them, by decreasing the volume of the first video, while simultaneously increasing the volume of the second video. The particular function "player.setVolume(volume: Number): Void" in the YouTube API allows this to be achieved for example.

Items (ii) and (iii) may be interchangeable, and only one or the other might be used. Item (iv) is required if the transition between tracks is to be performed using volume alterations.

Initially, the user has initiated play of the first video and starts playing the second in a separate media player. The algorithm implementing the embodiment of the invention will be notified that the script is playing by the "player state". A player state of "5" means that the video is paused, and a player state of "1" means it is playing. The algorithm will receive a message from the YouTube program containing state information such as "player state = 1"
which causes the algorithm to initiate the following process. Firstly the position of the video in milliseconds (e.g. "5.055 seconds from start") is sent in real time from the YouTube program application to the processor executing the algorithm. The algorithm compares the positions of both videos, and the nearest beat points to each video position, to see if the two videos are synced, with synchronisation being classed as any two beat points being located within 0.1 seconds of each other. For example, if beat points occur at 1.0 seconds and 1.5 seconds into tracks 1 and 2 respectively, and the videos were at 1.04 seconds and 1.45 seconds through playback respectively, that would be classed by the algorithm as synchronised, since the beat points are separated by 0.09 seconds. If the first and second videos were respectively at 1.06 and 1.44 seconds then each beat point would be off the current position by 0.06 seconds, with a total separation of 0.12 seconds, and so would not be classed as synchronised by the algorithm.

The algorithm does not need to wait until a beat point occurs in order to synchronise. If a beat point occurs at 1.45 seconds on track 1 and 1.90 seconds on track 2, then the tracks will be synchronised if track 1 is at 1.35 seconds and track 2 is 1.80 seconds at the same time. In other words, the algorithm can determine the relative difference at any one point and synchronise based on the relative difference since the positions of the beat points are contained in, or can be calculated from, metadata available to the algorithm, as described above for allowable difference.

Once the algorithm determines that the two videos are synced, it will let them both play. If no sync point is initially found then the second video is synchronised in any of the manners described herein. In particular, "stuttering" may be used whereby the second video is repeatedly paused temporarily and then quickly resumed. This serves to randomise and change the relative positions of the two videos' beat points relative to one another eventually making them closer together. After each "stutter", the algorithm compares times to see if the videos are synchronised. As mentioned above, this is done by comparing the relative positions received from the YouTube API. If no synchronisation is found the process is repeated.

Secondly, once synchronisation is established, the algorithm will start "cross-fading" the first track to the second track. Here, with the YouTube API, this can be accomplished by controlling the volume of each of the two tracks. The first track is faded out while the second track is faded in. Preferably this occurs over a length of 8 beats of the first track, although equally this could occur over 8 beats of the second track. The time period for the
fading can be calculated using the tempo of either the first or second track. For example, if the tempo is 60 BPM then each beat is 1 second long and therefore 8 beats has a duration of 8 seconds.

Since the YouTube API provides the position of videos to millisecond accuracy it is possible to work out the difference in position between two videos and to track this difference. It is also possible to control the play and pause of videos using the YouTube API. As such, any synchronisation determination techniques as described above can be used.

Figure 7 shows a remote computing device, such as a server 701, arranged to perform the methods described herein in conjunction with a local client user device 704. The server comprises a CPU 702 for performing the relevant calculations to determine the various tracks that can be blended. Other functional components, such as RAM, may be provided but are not shown for simplicity.

The CPU is coupled to a database, memory portion or store 711 for storing the collections of tracks for possible combination, as well as the associated metadata. The database may be stored on a common memory device, such as a hard disk drive or other type of storage device suitable for storing multimedia. Alternatively, collections of tracks may be stored in separate memories or stores to allow easier backup and updating/uploading of tracks. The metadata accompanying the tracks, which includes the timing data of beats or beat points for each track, may be stored on a separate memory within the server, or may even be stored in a different server accessible by the user device. The memory upon which the track database and/or metadata is stored is coupled to the CPU via a common bus.

The server system 701 further includes an input and output for receiving and sending data to other devices. The input/output 709 is shown as a common unit in Figure 7, but may be provided as separate interfaces. In either case, the server is preferably connected or connectable to a plurality of client user devices over a network such as the internet, or a local or private network. The network may be wired or wireless or a combination of both.

As shown in Figure 7, in communication with the server 701 is a user device 704 such as a computing device, laptop, tablet computer, smartphone or similar. The user device 704 may be in communication with the server 701 via a network connection such as the internet through which the user device can provide input to the server for selecting tracks to be blended. The network connection may alternatively be a local network, such as a home
network, with the server being a local media server. The server can also send data to the user device including indications of possible tracks that can be selected by the user for blending, as well as actually streaming the tracks to the user device. The user device 704 includes a CPU 705 for performing the necessary calculations to determine whether the first and second tracks are synchronised using metadata received from the server. The CPU may also calculate acceptable differences between the first and second tracks based on metadata from the server. The CPU is coupled to a display 706 which may or may not be integral to the user device, a user input device 707 for receiving input from the user, which again may or may not be integral to the user device (e.g., a touch screen, keyboard or mouse), and a memory 708 such as a hard disk drive for storing tracks for replay by the user device if required. The user device will also have an audio output device 710, such as a speaker, for outputting the tracks being streamed to the user. The audio output device may be integral to the user device 704, or may be separate from it.

It is possible that more than one server will be used to provide the tracks to a plurality of user devices. The servers streaming the media to the user device could be different to each other, and also different to the server used to provide the metadata information, such as beat/beat point locations. For example, embodiments could use up to three servers, or more, in total, e.g. music and live timing information are obtained from server 1 and server 2, and track metadata including beat point information, is provided from a database server.

The user device may be any suitable computing device such as a smartphone, tablet, PDA etc. It is also possible that any other type of computer, such as a desktop or laptop, may be used to implement the blending method.

According to certain embodiments of the invention the computer program that performs the synchronisation calculations, and initiates control of the media player controls in order to synchronise tracks, executes within a web browser program such as Internet Explorer or Firefox for example, or is associated therewith. It is from within the web browser that the media players are controlled and time differences are noted in order to determine synchronisation.

In the above description, the term “track” has been used to refer to audio files containing a piece of music or a song. The audio files may be in any appropriate format such as AIFF, WAV, FLAC, WMA, RealAudio, MP3 or any file type that may accompanying a video file.
In addition to the tempo filtering it was mentioned above that some sort of key filtering can be applied. The key of each track in the track database is preferably determined automatically using an appropriate software algorithm configured to extract the key information from the audio file. Once the keys of the two tracks to be combined are determined, a comparison is performed to determine whether the keys are matching (the same key) or otherwise related in a musically pleasing way. Preferably, related keys are determined by using the “circle of fifths” (sometimes also known as the “circle of fourths”). The circle of fifths shows the relationship between the twelve tones of the chromatic scale and their corresponding key signatures. An example is shown in Figure 8, which is preferably converted into database form for use in embodiments of the invention.

For a given key, the adjacent key in the adjacent circle will be related, so the key of F# Minor is related to the key of A Minor. In addition, for a given key the adjacent keys within the same circle are also related, so the key of F# Minor is related to the keys of B Minor and C# Minor. In addition, the keys next to the adjacent key in the adjacent circle are also related, so the key of F# Minor is also related to the keys of D and E. These keys are highlighted with the thick black line in Figure 8. Therefore, if a track is in the key of F# Minor it may be combined with tracks that are either in the key of F# Minor, A, D, E, B Minor or C# Minor. It is most preferable to combine tracks that are within the same, or matching, keys but musically pleasing results may be obtained by combining with the other related keys. The user may be presented with the option of using tracks that only match exact keys or with tracks that are also in the related keys.

Embodiments of the invention can be implemented as a distributed system involving a server and one or more user devices that access audio tracks stored at the server. The calculations involved in synchronising two tracks may be performed locally on the user device based on metadata received from the server, or they may be performed remotely at the server, or any combination of steps may be performed at the server or the user device.
CLAIMS

1. A computer implemented method of transitioning from a first audio track to a second audio track, the method comprising, using a processor:

   - controlling a first media player to play the first audio track at a user device and output the first audio track from an audio output device associated with the user device;
   - controlling the first media player or a second media player to simultaneously play the second audio track at the user device without outputting the second audio track from the audio output device;
   - determining whether the first audio track and the second audio track are synchronised and, if they are not synchronised implementing a synchronisation procedure comprising:
     i) controlling the first media player or the second media player to adjust the second audio track to alter the position of the second audio track relative to the first audio track by pausing the second track for a period of time;
     ii) checking whether the first audio track and second audio track are synchronised;
     iii) repeating steps (i) and (ii) until the first and second audio track are synchronised;
   - the method further comprising controlling the first media player and/or the second media player to transition from outputting the first audio track from the audio output device to outputting the second audio track from the audio output device.

2. A method according to claim 1 wherein controlling the media player to pause the second audio track for a period of time comprises pausing the second audio track for a predetermined period of time.

3. A method according to claim 2 wherein the period of time is between 300 and 600 milliseconds, or between 100 and 300 milliseconds.
4. A method according to claim 1 wherein controlling the media player to pause the second audio track for a period of time comprises pausing the second audio track for a random period of time.

5. A method according to claim 4 wherein the period of time is randomly selected from the range between 300 and 600 milliseconds, or between 100 and 300 milliseconds.

6. A method according to any preceding claim further comprising the step of receiving user input to select the second audio track from a subset of audio tracks, the subset of audio tracks being selected from a database of audio tracks based on track tempo, wherein the tempo of each audio track in the subset is within a predetermined range of the first audio track.

7. A method according to claim 6 wherein the tempo of each audio track in the subset is within 3 BPM of the first audio track.

8. A method according to claim 7 wherein the tempo of each audio track in the subset is within 1 BPM of the first audio track.

9. A method according to any preceding claim, the method further comprising periodically receiving timing information indicative of the time reached within the first track and the time reached within the second track, wherein the steps of determining and/or checking whether the first audio track and the second audio track are synchronised is based on comparisons of the time of the first track and the time of the second track.

10. A method according to claim 9 wherein determining and/or checking whether the first audio track and the second audio track are synchronised comprises determining, or receiving data indicative of, the timing location of beats within the tracks, and comparing the timing location of beats within the tracks.

11. A method according to claim 9 or 10 wherein synchronisation of the first and second audio tracks occurs when the relative timings of the first and second track are such that a beat in the first track is within a predetermined period of time of a beat in the second track.

12. A method according to claim 10 or 11 further comprising comparing the timing location of beats within the first track with the timing location of beats in the second track to
determine acceptable timing differences between the first and second tracks such that
when the time reached by the tracks are separated by an acceptable timing difference they
are synchronised.

13. A method according to claim 12 wherein the acceptable timing differences
correspond to the difference between the timing location of the n\textsuperscript{th} beat of first track and the
n\textsuperscript{th} beat of the second track plus the length of a beat in the first or second track multiplied
by y, where n and y may be any positive integer.

14. A method according to any of claims 10 to 13 wherein the data indicative of the
timing location of beats within the tracks is determined or received for beat points within the
tracks, the beat points being a location within each track that corresponds to the
occurrence of a particular beat within a sequence of beats,

15. A method according to claim 14 wherein the beat point corresponds to every 4\textsuperscript{th} or
8\textsuperscript{th} beat.

16. A method according to any preceding claim wherein the second audio track is
initially muted, the step of transitioning from the first audio track to the second audio track
comprising controlling the first media player to fade the first audio track out and controlling
the first or second media player to simultaneously fade the second audio track in.

17. A method according to any preceding claim wherein controlling the media player or
media players to transition from the first audio track to the second audio track occurs at the
same time as the synchronisation procedure.

18. A method according to any of claims 1 to 16 wherein controlling the media player or
media players to transition from the first audio track to the second audio track occurs after
the synchronisation procedure.

19. A method according to any of claims 16 or 17 wherein the transition from the first
audio track to the second audio track comprises decreasing the volume of the first audio
track from a first level to a second level over a transition period and simultaneously
increasing the volume of the second audio track from a third level to a fourth level over the
transition period.
20. A method according to claim 19 wherein the second and third levels are zero volume.

21. A method according to claim 19 or 20 wherein the transition period corresponds to 8 beats of the first track or second track.

22. A method according to any preceding claim wherein the audio tracks are received by streaming from a server device over a network.

23. A computer program which when executed on a computer causes it to undertake the method of any preceding claim.

24. A user device (704) comprising a processor, the user device comprising, or being configured to be coupled to, an audio output device (710), the user device further comprising a communication interface for communicating with a server (701) over a network, the processor being configured to:

- receive a first audio track from the server;
- control a first media player, executing on the user device, to play the first audio track and output the first audio track from the audio output device;
- receive a second audio track from the server;
- control the first media player or a second media player, executing on the user device, to simultaneously play the second audio track without outputting the second audio track from the audio output device;
- determine whether the first audio track and the second audio track are synchronised and, if they are not synchronised implementing a synchronisation procedure comprising:
  i) controlling the first media player or the second media player to adjust the second audio track to alter the position of the second audio track relative to the first audio track by pausing the second track for a period of time;
  ii) checking whether the first audio track and second audio track are synchronised;
  iii) repeating steps (i) and (ii) until the first and second audio track are synchronised;
- the processor being further configured to control the first media player and/or the second media player to transition from outputting the first audio
track from the audio output device to outputting the second audio track from the audio output device.

25. A user device according to claim 24, the processor being further configured to carry out the method of any of claims 2 to 22.

26. A user device according to claim 24 or 25 further comprising a user input device (707) for receiving user input selecting audio tracks to be received from the server, the processor being configured to request the tracks from the server in response to the user input.

27. A user device according to any of claims 24 to 26 wherein the media players are executed within a web browser program.

28. A user device according to any of claims 24 to 27 wherein the computer program to cause the processor to implement the synchronisation procedure is implemented as a web browser based script.

29. A user device according to any of claims 24 to 28 wherein the user device is a computer, laptop, tablet, smartphone, PDA or other computing device.

29. A server comprising a processor (702), an audio track database (711) and a communication interface (709) for communicating with client user devices (704) over a network such as the internet, the processor being configured to:

- send data to a user device indicative of tracks that can be selected from the audio track database;
- receive a selection of a first track from the user device and stream the first track to the user device; and
- receive a selection of a second track from the user device and stream the second track to the user device simultaneously with the first track.

30. A server according to claim 29 wherein the processor is further configured to send, with the first and second tracks, accompanying timing data for each track.
31. A server according to claim 29 or 30 wherein the server further stores metadata for each track indicative of the location of the timing location of beats within the first track and timing location of beats in the second track.

32. A server according to claim 31 wherein the metadata indicative of the timing location of beats within the tracks is pre-determined for beat points within the tracks, the beat points being a location within each track that corresponds to the occurrence of a particular beat within a sequence of beats.

33. A user device substantially as herein described with reference to the accompanying Figures.
AMENDMENTS TO THE CLAIMS FILED AS FOLLOWS:–

CLAIMS

1. A computer implemented method of transitioning from a first audio track to a second audio track, the method comprising, using a processor:
   - controlling a first media player to play the first audio track at a user device and output the first audio track from an audio output device associated with the user device;
   - controlling the first media player or a second media player to simultaneously play the second audio track at the user device without outputting the second audio track from the audio output device;
   - determining whether the first audio track and the second audio track are synchronised and, if they are not synchronised implementing a synchronisation procedure comprising:
     i) controlling the first media player or the second media player to adjust the second audio track to alter the position of the second audio track relative to the first audio track by pausing the second track for a period of time;
     ii) checking whether the first audio track and second audio track are synchronised;
     iii) repeating steps (i) and (ii) until the first and second audio track are synchronised;
   - the method further comprising controlling the first media player and/or the second media player to transition from outputting the first audio track from the audio output device to outputting the second audio track from the audio output device.

2. A method according to claim 1 wherein controlling the media player to pause the second audio track for a period of time comprises pausing the second audio track for a predetermined period of time.

3. A method according to claim 2 wherein the period of time is between 300 and 600 milliseconds, or between 100 and 300 milliseconds.
4. A method according to claim 1 wherein controlling the media player to pause the second audio track for a period of time comprises pausing the second audio track for a random period of time.

5. A method according to claim 4 wherein the period of time is randomly selected from the range between 300 and 600 milliseconds, or between 100 and 300 milliseconds.

6. A method according to any preceding claim further comprising the step of receiving user input to select the second audio track from a subset of audio tracks, the subset of audio tracks being selected from a database of audio tracks based on track tempo, wherein the tempo of each audio track in the subset is within a predetermined range of the first audio track.

7. A method according to claim 6 wherein the tempo of each audio track in the subset is within 3 BPM of the first audio track.

8. A method according to claim 7 wherein the tempo of each audio track in the subset is within 1 BPM of the first audio track.

9. A method according to any preceding claim, the method further comprising periodically receiving timing information indicative of the time reached within the first track and the time reached within the second track, wherein the steps of determining and/or checking whether the first audio track and the second audio track are synchronised is based on comparisons of the time of the first track and the time of the second track.

10. A method according to claim 9 wherein determining and/or checking whether the first audio track and the second audio track are synchronised comprises determining, or receiving data indicative of, the timing location of beats within the tracks, and comparing the timing location of beats within the tracks.

11. A method according to claim 9 or 10 wherein synchronisation of the first and second audio tracks occurs when the relative timings of the first and second track are such that a beat in the first track is within a predetermined period of time of a beat in the second track.

12. A method according to claim 10 or 11 further comprising comparing the timing location of beats within the first track with the timing location of beats in the second track to
determine acceptable timing differences between the first and second tracks such that when the time reached by the tracks are separated by an acceptable timing difference they are synchronised.

13. A method according to claim 12 wherein the acceptable timing differences correspond to the difference between the timing location of the nth beat of first track and the nth beat of the second track plus the length of a beat in the first or second track multiplied by y, where n and y may be any positive integer.

14. A method according to any of claims 10 to 13 wherein the data indicative of the timing location of beats within the tracks is determined or received for beat points within the tracks, the beat points being a location within each track that corresponds to the occurrence of a particular beat within a sequence of beats.

15. A method according to claim 14 wherein the beat point corresponds to every 4th or 8th beat.

16. A method according to any preceding claim wherein the second audio track is initially muted, the step of transitioning from the first audio track to the second audio track comprising controlling the first media player to fade the first audio track out and controlling the first or second media player to simultaneously fade the second audio track in.

17. A method according to any preceding claim wherein controlling the media player or media players to transition from the first audio track to the second audio track occurs at the same time as the synchronisation procedure.

18. A method according to any of claims 1 to 16 wherein controlling the media player or media players to transition from the first audio track to the second audio track occurs after the synchronisation procedure.

19. A method according to any of claims 16 or 17 wherein the transition from the first audio track to the second audio track comprises decreasing the volume of the first audio track from a first level to a second level over a transition period and simultaneously increasing the volume of the second audio track from a third level to a fourth level over the transition period.
20. A method according to claim 19 wherein the second and third levels are zero volume.

21. A method according to claim 19 or 20 wherein the transition period corresponds to 8 beats of the first track or second track.

22. A method according to any preceding claim wherein the audio tracks are received by streaming from a server device over a network.

23. A computer program which when executed on a computer causes it to undertake the method of any preceding claim.

24. A user device (704) comprising a processor, the user device comprising, or being configured to be coupled to, an audio output device (710), the user device further comprising a communication interface for communicating with a server (701) over a network, the processor being configured to:
   - receive a first audio track from the server;
   - control a first media player, executing on the user device, to play the first audio track and output the first audio track from the audio output device;
   - receive a second audio track from the server;
   - control the first media player or a second media player, executing on the user device, to simultaneously play the second audio track without outputting the second audio track from the audio output device;
   - determine whether the first audio track and the second audio track are synchronised and, if they are not synchronised implementing a synchronisation procedure comprising:
     i) controlling the first media player or the second media player to adjust the second audio track to alter the position of the second audio track relative to the first audio track by pausing the second track for a period of time;
     ii) checking whether the first audio track and second audio track are synchronised;
     iii) repeating steps (i) and (ii) until the first and second audio track are synchronised;
   - the processor being further configured to control the first media player and/or the second media player to transition from outputting the first audio
track from the audio output device to outputting the second audio track from the audio output device.

25. A user device according to claim 24, the processor being further configured to carry out the method of any of claims 2 to 22.

26. A user device according to claim 24 or 25 further comprising a user input device (707) for receiving user input selecting audio tracks to be received from the server, the processor being configured to request the tracks from the server in response to the user input.

27. A user device according to any of claims 24 to 26 wherein the media players are executed within a web browser program.

28. A user device according to any of claims 24 to 27 wherein the computer program to cause the processor to implement the synchronisation procedure is implemented as a web browser based script.

29. A user device according to any of claims 24 to 28 wherein the user device is a computer, laptop, tablet, smartphone, PDA or other computing device.

30. A user device substantially as herein described with reference to the accompanying Figures.
Applicant No: GB1217381.1
Examiner: Rhiannon Jenkins
Claims searched: 1-29
Date of search: 25 January 2013

Patents Act 1977: Search Report under Section 17

Documents considered to be relevant:

<table>
<thead>
<tr>
<th>Category</th>
<th>Relevant to claims</th>
<th>Identity of document and passage or figure of particular relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>1-29</td>
<td>US 2012/046954 A1 (APPLE INC) - See paragraphs [0025] to [0051]</td>
</tr>
<tr>
<td>X</td>
<td>1-29</td>
<td>US 2004/254660 A1 (SEEFELDT) - See figure 6 and paragraphs [0049] to [0102]</td>
</tr>
<tr>
<td>A</td>
<td>-</td>
<td>US 2006/000344 A1 (MICROSOFT CORP) - See paragraph [0008]</td>
</tr>
</tbody>
</table>

Categories:

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Document indicating lack of novelty or inventive step</td>
<td>A</td>
</tr>
<tr>
<td>Y</td>
<td>Document indicating lack of inventive step if combined with one or more other documents of same category.</td>
<td>B</td>
</tr>
<tr>
<td>&amp;</td>
<td>Member of the same patent family</td>
<td>C</td>
</tr>
</tbody>
</table>

Field of Search:

Search of GB, EP, WO & US patent documents classified in the following areas of the UKC:

Worldwide search of patent documents classified in the following areas of the IPC:

G06F; G10H; G10L; G11B

The following online and other databases have been used in the preparation of this search report:

EPODOC, WPI, INTERNET

International Classification:

<table>
<thead>
<tr>
<th>Subclass</th>
<th>Subgroup</th>
<th>Valid From</th>
</tr>
</thead>
<tbody>
<tr>
<td>G10H</td>
<td>0001/36</td>
<td>01/01/2006</td>
</tr>
<tr>
<td>G06F</td>
<td>0017/30</td>
<td>01/01/2006</td>
</tr>
<tr>
<td>G11B</td>
<td>0027/038</td>
<td>01/01/2006</td>
</tr>
</tbody>
</table>