Visual Speech Synthesis by Morphing Visemes

顔画像がしゃべるテキスト読上げシステム

Tony Ezzat † Tomaso Poggio †

Abstract

We present MikeTalk, a text-to-audiovisual speech synthesizer which converts input text into an audiovisual speech stream. MikeTalk is built using *visemes*, which are a set of images spanning a large range of mouth shapes. The visemes are acquired from a recorded visual corpus of a human subject which is specifically designed to elicit one instantiation of each viseme. Using optical flow methods, *correspondence* from every viseme to every other viseme is computed automatically. By *morphing* along this correspondence, a smooth transition between viseme images may be generated. A complete visual utterance is constructed by *concatenating* viseme transitions. Finally, phoneme and timing information extracted from a text-to-speech synthesizer is exploited to determine which viseme transitions to use, and the rate at which the morphing process should occur. In this manner, we are able to synchronize the visual speech stream with the audio speech stream, and hence give the impression of a photorealistic talking face.

あらまし

MikeTalkは，テキストを入力すると顔画像がしゃべりながら音声で読み上げるシステムである．様々な口形状を表現するためにヴァイジーム（viseme）と呼ぶ画像列集合を用いる．まず，各ヴァイジームが1回は含まれるように被験者に発話してもらった画像コーパスを記録し，その画像コーパスからヴァイジームを作成する．次に，すべてのヴァイジームの対について，あるヴァイジームから他のヴァイジームへ対応づけができるように，オプティカル・フローと呼ぶ画像処理手法を用いる．さらに，この対応づけ結果にモーフィング手法を適用してヴァイジーム間で違和感のない自然なつながりを持った発話画像を生成する．最後に，テキストに対する音声合成処理から得られた音素とそのタイミングを用いて，つなぎ合わせるヴァイジームとモーフィングの速度を決定することによって，写真のようにリアルに話す顔画像と音声が同期してテキストを読み上げる．

1 Introduction

The goal of the work described in this paper is to develop a text-to-audiovisual speech synthesizer (TTVS) called MikeTalk. MikeTalk is similar to a standard text-to-speech synthesizer in that it converts text into an audio speech stream. However, MikeTalk also produces an accompanying visual stream composed of a talking face enunciating that text.

In this work, we are particularly interested in building a TTVS system where the facial animation is *video-realistic*: that is, we desire our talking facial model to look as much as possible as if it were a videocamera recording of a human subject, and not that of a cartoon-like human character.

In addition, we choose to focus our efforts on the issues related to the synthesis of the visual speech stream, and not on audio synthesis. For the task of converting text to audio, we have incorporated into our work the Festival speech synthesis system, which was developed by Alan Black, Paul Taylor, and colleagues at the University of Edinburgh.[1)]

The Festival TTS system, as with most speech synthesizers, divides the problem of converting text to speech into two sub-tasks: first, a natural language processing (NLP) unit converts the input text into a set of output streams that contain relevant phonetic, timing, and other international parameters; second, an audio signal processing unit converts the NLP output streams into an audio stream in which the input text is enunciated.

† Center for Biological and Computational Learning, MIT

Within this framework, our goal in this work, is two-fold: first, to develop a *visual speech module* that takes as input the phonetic and timing output streams generated by Festival's NLP unit, and produces as output a visual speech stream of a face enunciating the input text. Secondly, to develop a *lip-sync module* that synchronizes the playback of the audio and visual streams.

We discuss the facial animation module in Sections 2 through 4, and the lip-sync module in Section 5.

2 Corpus and Viseme Data Acquisition

The basic underlying assumption of our facial synthesis approach is that the complete set of mouth shapes associated with human speech may be reasonably spanned by a finite set of *visemes*. In this work, we define a viseme to be a static lip shape image that is visually contrastive from another.

Given the assumption that visual speech is spanned by a set of visemes, we assume that there exists a *one-to-one mapping* between the set of phonemes and the set of visemes, and design the corpus so that there is at least one word uttered which instantiates each phoneme.

Our recorded corpus is shown in **Figure 1**. The example words uttered are obtained from reference (2), and are generally categorized into example words which instantiate consonantal, monophthong vocalic, and diphthong vocalic phonemes. After the whole corpus is recorded and digitized, one lip image is extracted as an instance of that viseme. In general, the viseme image extracted was chosen as the image occurring at the point where the lips were judged to be at their *extremal* position for that sound.

The one-to-one mapping strategy thus leads to the extraction of 52 viseme images in all: 24 representing the consonants, 12 representing the monophthongs, and 16 representing the diphthongs.

Since a large number of the extracted visemes looked similar, it was decided to further reduce the viseme set by grouping them together. This was done in a subjective manner, by comparing the viseme images visually to assess their similarity. The final reduced set of visemes are shown in **Figures 2** and **3**. Note that while our figures display only the region around the mouth, our viseme imagery capture the entire face. *In all, there are 16 final visemes*. Six visemes represent the 24 consonantal phonemes. Seven visemes represent the 12 vocalic phonemes.

3 Viseme Morphing

In constructing a visual speech stream, it is not sufficient to simply display the viseme images in sequence. Doing so would create the disturbing illusion of very abrupt mouth movement, since the viseme images differ significantly from each other in shape. Consequently, a mechanism of transitioning from each viseme image to every other viseme image is needed, and this transition must be smooth and realistic. In this work, a *morphing* technique was adopted to create this transition.

As a first step, all morphing methods require the specification of *correspondence* maps $C_0 : I_0 \Rightarrow I_1$ and $C_1 : I_1$

monophthongs		consonants	
ii	bead	r	ride
i	bid	l	light
e	bed	w	wide
a	bad	y	yacht
o	body	m	might
aa	father	n	night
uh	bud	ng	song
oo	baud	b	bite
u	book	d	dog
uu	boot	g	get
@	about	p	pet
@@	bird	t	tea
		k	key
diphthongs		v	veal
		dh	then
ou	boat	z	zeal
ei	bait	zh	garage
au	bout	f	feel
ai	bide	th	thin
oi	boyd	s	seal
e@	there	sh	shore
i@	near	h	head
u@	moor	jh	jeep
		ch	chore

Figure 1. Recorded visual corpus. The underlined portion of each example word identifies the target phoneme being recorded. To the left of each example word is the phonemic transcription label being used.

図 1． 画像収録に用いたコーパス．各単語において，録画対象の音素を下線で示す．また，各単語の左の文字は，音素表現ラベルである．

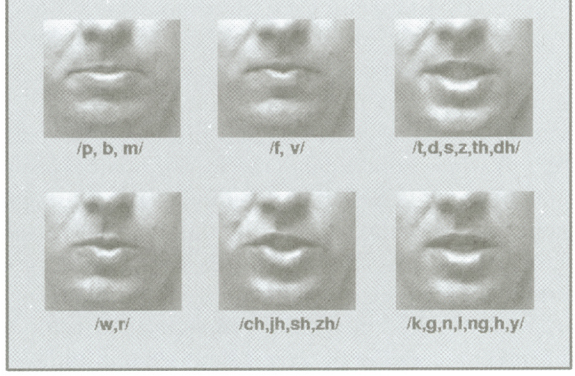

Figure 2. Six consonant visemes.

図 2． 子音ヴァイジーム（6 種）．

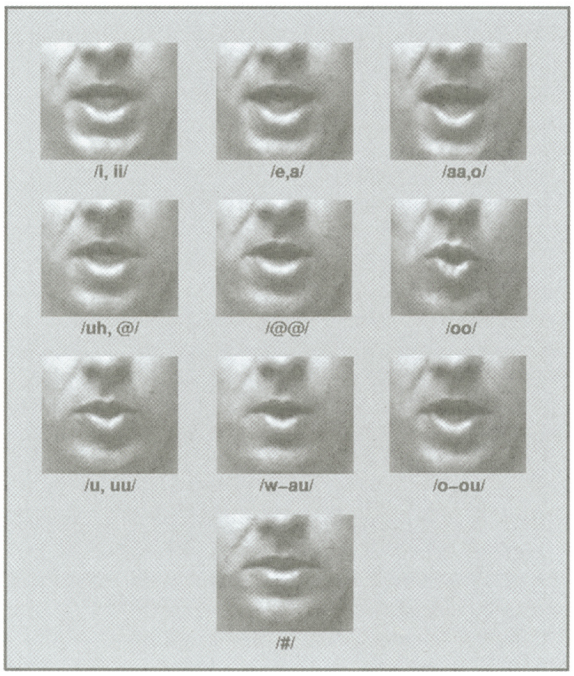

Figure 3. Seven monophthong visemes, 2 diphthong visemes, and the silence viseme.

図 3 . 単母音ヴァイジーム（7 種）と二重母音ヴァイジーム（2 種）と無言ヴァイジーム.

⇒I_0 relating the images I_0 and I_1 to each other. We compute these correspondence maps automatically using optical flow. Optical flow[3] was originally formulated in the context of measuring the apparent motion of objects in images. Given two images I_0 and I_1, computing optical flow with I_0 as reference image produces a correspondence map C_0, while computing optical flow with I_1 as reference produces a correspondence map C_1. In this work, we utilize the *coarse-to-fine, gradient-based* optical flow algorithms developed by reference (4).

After computing the correspondence map C_0 between two viseme images I_0 and I_1, our morphing algorithm then *forward warps* I_0 along that correspondence. Forward warping may be viewed as "pushing" the pixels of I_0 along the computed flow vectors. By scaling the flow vectors uniformly by a parameter α between 0 and 1, one can produce a series of warped intermediate images $I_0^{warped}(\alpha)$ which approximate the transformation between visemes I_0 and I_1. Several such intermediate warps are shown in **Figure 4** (a), where I_0 is the \m\ viseme and I_1 is the \aa\ viseme. The black *holes* which appear in the intermediate images occur in cases where a destination pixel was not filled in with any source pixel value. The main reason for this is that the computed optical flow displacements exhibit nonzero divergence, particularly around the region where the mouth is expanding.

In symmetric fashion, it is also possible to forward warp

Figure 4. (a) Forward warping viseme I_0 (first image) towards I_1. (b) Forward warping viseme I_1 (last image) towards I_0. (c) Morphing the images in I_0 and I_1 together. (d) The same morphed images as in (c), after hole-filling and median-filtering. Note that our morphing algorithm operates on the entire facial image, although we only show the region around the mouth for clarity.

図 4 . (a) I_0（左端画像）からI_1への前向きワープ時のヴァイジーム．(b) I_1（右端画像）からI_0への後向きワープ時のヴァイジーム．(c) I_0とI_1のモーフィングを同時に行った場合．(d)(c)と同様にモーフィングした後に，穴埋め補間とメディアン・フィルタを行った画像．なお，ここでは，口周辺のみを示したが，実験では顔全体に対してモーフィングを行っている．

I_1 towards I_0 along reverse correspondence map C_1 from I_1 towards I_0. Several intermediate images of this reverse forward warp are shown in Figure 4 (b).

Because forward warping can only move pixels around, *it cannot model the appearance of new pixel texture*. As is evident from the sequence in Figure 4 (a), a forward warp of viseme I_0 along the flow vectors of C_0 can never produce a final image that looks like viseme I_1, since viseme I_1 itself contains a large amount of novel texture from the inside of the mouth.

Morphing overcomes this "novel pixel texture" problem by combining the texture found in both forward warps. This combination is performed by scaling the warped intermediate images with respective *cross-dissolve* or *blending* parameters, and then adding to produce the final morphed image $I^{morph}(\alpha)$. By interpolating the blending parameters the morph "fades out" the warped versions of the starting viseme and"fades in"the warped versions of the ending viseme. The blending process thus allows the two warps to be effectively combined, and the "new" pixels of the second viseme to become involved in the viseme transition itself.

As may be seen in the images of Figure 4 (c), there are locations for which *neither* warp predicts a pixel value, leaving a set of visible holes in the morphed images. To

remedy this, a hole-filling algorithm proposed in reference (5) was adopted. The algorithm traverses the destination image in scanline order and fills in the holes by interpolating linearly between their non-hole endpoints. This approach works reasonably well whenever the holes are small in size, which is the case here. Figure 4 (d) shows the same set of morphed intermediates as in Figure 4 (c), but with the holes filled and median-filtered.[6]

Overall, we have found that the above morphing approach produces remarkably realistic transitions between a wide variety of viseme imagery, including the typically "hard" morph transitions between open and closed mouths shown in Figure 4.

4 Morph Concatenation

To construct a visual stream in which a word or a sentence is uttered, we simply *concatenate* the appropriate viseme morphs together. For example, the word one, which has a phonetic transcription of \w-uh-n\, is composed of the two viseme morphs \w-uh\ and \uh-n\ put together and played seamlessly one right after the other.

5 Audiovisual Synchronization

As discussed earlier, we have incorporated the Festival TTS system[1] into our work. In a manner that is completely analogous to our method for concatenating viseme morphs, Festival constructs the final audio speech stream by concatenating *diphones* together.

In order to produce a visual speech stream in synchrony with the audio speech stream, our lip-sync module first extracts the duration of each diphone as computed by the audio module. Next, the lip-sync module creates an intermediate stream, called the *viseme transition* stream. A viseme transition is defined to be the collection of two endpoint visemes and the optical flow correspondences between them. Thirdly, the lip-sync module creates the *video stream*, which is composed of a sequence of *frames* which sample the chosen viseme transitions. The lip-sync algorithm determines how to synthesize each frame F_k by setting the morph parameter α_k for that frame to be the ratio between the time elapsed from the start of the nearest viseme transition to the frame itself, divided by the entire duration of the nearest viseme transition.

As a final step, each frame is synthesized using the morph algorithm discussed in Section 3.

6 Summary

In summary, our talking facial model may be viewed as a *collection of viseme imagery and the set of optical flow vectors defining the morph transition paths from every viseme to every other viseme.*

References

* Our results may be viewed by accessing our World Wide Web home page at http://cuneus.ai.mit.edu:8000/research/miketalk/miketalk.html. We have synthesized several audiovisual sentences to test our overall approach for visual speech synthesis and audio synchronization described in the text. The first author may also be contacted for a video tape which depicts the results of this work.

(1) A. Black and P. Taylor: "The Festival Speech Synthesis System," University of Edinburgh, 1997.

(2) J. Olive, A. Greenwood, and J. Coleman: "Acoustics of American English Speech: A Dynamic Approach," Springer-Verlag, New York, USA, 1993.

(3) B. K. P. Horn and B. G. Schunck: "Determining optical flow," Artificial Intelligence, Vol. 17, pp. 185-203, 1981.

(4) J. R. Bergen and R. Hingorani: "Hierarchical motion-based frame rate conversion," Technical report, David Sarnoff Research Center, Princeton, New Jersey, April 1990.

(5) Shenchang Eric Chen and Lance Williams: "View interpolation for image synthesis," In SIGGRAPH'93 Proceedings, pp. 279-288, Anaheim, CA, August 1993.

(6) J. Lim: "Two-dimensional signal and image processing," Prentice Hall, Englewood Cliffs, New Jersey, 1990.

Author Biographies

Tony Ezzat is currently a doctoral candidate at the Department of Computer Science at MIT, where his research interests include computer vision, computer graphics, audiovisual speech processing, and compression. He recieved the BS in Electrical Engineering, BS in Computer Science, and MEng in Computer Science, all in 1996 from MIT.

Tomaso Poggio is currently the Uncas and Helen Whitaker Professor of Vision Sciences and Biophysics at the Department of Brain and Cognitive Sciences (BCS) at MIT, and is also affiliated with MIT's Artifical Intelligence Laboratory. In addition, he is Co-Director of MIT's Center for Biological and Computational Learning (CBCL) since 1993. He received a Ph.D. in Theoretical Physics from the University of Genoa in 1970, and his current research focuses primarily on the application of new learning techniques to time series analysis, object recognition, adaptive control and computer graphics.