# Wakey-Wakey: 通过模仿 GIF 中的角色来使文本动画化

**摘要**

由于具有吸引人的视觉效果，动态排版（动画文本）在电影、广告和社交媒体中盛行。然而，制作动画方案仍然具有挑战性和耗时。我们提出了一个自动框架，将给定表情包 GIF 上刚体的动画方案转移到矢量格式的文本上。首先，提取 GIF 锚点上关键点的轨迹，并基于局部仿射变换将其映射到文本的控制点；然后优化控制点的时间位置以保持文本拓扑结构。本文还开发了一个创作工具，允许在生成过程中进行直观的人工控制。一项问卷调查研究提供了证据，证明输出结果美观，并且很好地保留了原始 GIF 中的动画模式，参与者对原始 GIF 的类似情感语义印象深刻。

**关键词：**动画文本；局部仿射变换

## 1 引言

由于我的研究方向是情感可视化，而文本动画在传达情感方面有着重要的应用，因此我想探索动画文本方向的一些研究，从而选择这篇论文作为我的复现论文；如今，动态排版，即动画文本或运动文本，已经在日常生活中变得普遍。它的应用场景，包括：动画可视化 [17]、即时通讯 [8,14]、环境显示 [13] 和字幕 [11]。然而，为文本元素制作动画仍然很重要。利用商业动画软件 [1,2] 或编程工具包 [31]，虽然可以调整每个动画关键帧中的文本配置，但是有太多的低级参数需要仔细考虑，设计空间过大，导致这一过程仍然耗时且具有挑战性。本文受人工智能最新进展的启发，头部图像可以通过模仿驾驶视频的动作来说话，例如 [9,19]，从而探索将现有的动画设计转换为文本。现有的方法并不直接适用于我们的目标。一方面，运动迁移的研究很少关注非真实感领域 [15,16]，特别是对于动态排版。另一方面，文本风格化的相关研究主要集中在静态文本上 (例如 [10,18])，而动画的研究在很大程度上还没有得到充分的探索。因此本文提出了一个混合倡议框架，用于创建基于带有移动字符的驱动 GIF 的动态排版。在机器端，驱动 GIF 的运动被表示为几个关键点的轨迹，这些关键点被提取出来并引导目标文本控制点的位置变化。在人类方面，人们可以通过直接操纵这些点来控制映射过程，以细化每个点的自动计算位置，从而产生更理想的输出。基于提出的框架，本文开发了一个用于创建动态排版的交互界面。

## 2 相关工作

### 2.1 数字版式的初探

字体是一组经过设计的字符或字母，命名为字形，如 Courier New、Times New Roman 和 Bookman Old。在计算机中，字体以栅格域或矢量域表示；由于位图字体在高分辨率下可能会出现像马赛克一样的锯齿状边缘，因此我们选择了一种基于矢量的字体——TrueType [3] 字体，它以二次贝塞尔曲线来描述字形。TrueType 字体是基于矢量图形的，这意味着它们可以无损缩放到任何大小而不失真。这对于在不同大小的设备和分辨率上显示文本非常有用。TrueType 允许字体设计者更精确地控制字形的形状和轮廓，以确保在各种大小和分辨率下都能呈现出高质量的字形。

### 2.2 动态排版

动态排版丰富了动画用户界面 [5] 和数字媒体，自 20 世纪 90 年代以来一直受到学术关注 [7]。与静态文本相比，动态排版更能引导注意力 [4, 13]，并通过动画背后的副语言线索传达情感或语义 [11, 12]。

### 2.3 引导动画生成

作者的目的是给定一个静态文本，要生成一个情感和语义上相符的动态文字；一些相关工作是根据源域的一些特征去推理生成下一个动作，而文字并没有这些特征，故无法应用于基于矢量的文本上；因此，采用了 FOMM 模型，这是最先进的运动转移模型之一，原理是将源和目标域提取的关键点对齐的方式对任意物体进行动作转移，并且在提取的关键点进行局部微调变形的操作。具体来说，我们通过正则化锚点与拉普拉斯坐标的距离变化来维持文本的可读性。这个想法与 DAM 模型去维持结构信息的主意一致，引入了一个潜在的根锚点来模拟物体的结构。与作者对文本的关注不同，大多数现有的数据集和模型都关注说话的头和人类的姿势，本文对动态排版的探索有助于跨域运动转移的独特案例。

## 3 本文方法

### 3.1 本文方法概述

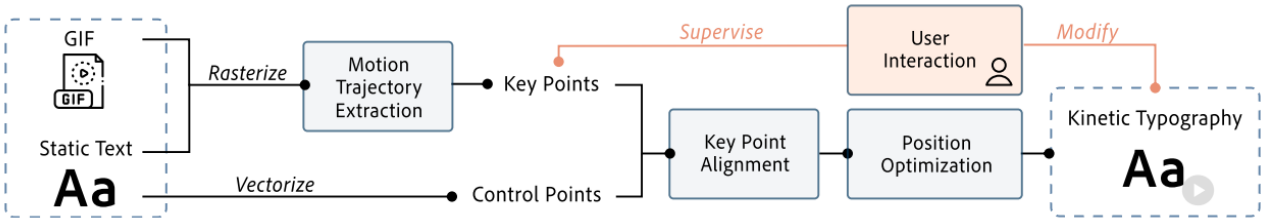


图 1. Overview of the approach

图1显示我们方法的一个框架 [6]。首先输入是以 TrueType 格式表示的文本，以及一个带有锚点的 GIF 动图，文本转换为 image 后与 GIF 一起输入到 FOMM 模型，从而提取出 GIF

每个帧中关键点的运动轨迹 Xfj，同时也会将矢量化的静态文本解析成一个控制点集 C0, 然后对 X 和 C 进行局部仿射变形（关键点对齐）从而得到一个文本控制点的运动轨迹 $C_j^f$，然后对这个轨迹进行位置优化得到 $C'^f_j$ ，形成一个矢量化的字形序列，最后生成一个矢量化的动画文本。作者开发了一个创作工具，允许在生成过程中进行直观的人工控制。用户可以通过控制两个超参数来控制 $X_j^f$ 和 $C'^f_j$ 的生成，从而微调生成的动画文本，用户还可以手动调整不合理的关键点的位置，以及精细的调整文本控制点，从而对动画文本逐帧的细粒度的细化。

### 3.2 运动轨迹提取

本文采取的方法是将静态文本转换为图像，并将其与锚定 GIF 一起输入，以获得运动关键点的轨迹。由于 GIF 中的对象通常与文本的形状不同，因此需要从源 GIF 中分离外观并提取关键点的运动轨迹。本文使用 FOMM[45] 进行关键点提取。它是一种自监督的方法，使用了外观和运动解耦的框架，有效地丰富了可能的可转移运动，以支持任何对象类别内的运动转移。为了解决驱动帧 D 与源图像 S 在关键点上存在较大差异的问题，FOMM 模型引入了抽象参考帧 R，分别计算 $\mathcal{T}_{S\leftarrow R}$ 和 $\mathcal{T}_{D\leftarrow R}^{-1}$ 得到 $\mathcal{T}_{S\leftarrow D}$。

$$\mathcal{T}_{S\leftarrow D} = \mathcal{T}_{S\leftarrow R} \circ \mathcal{T}_{R\leftarrow D} = \mathcal{T}_{S\leftarrow R} \circ \mathcal{T}_{D\leftarrow R}^{-1} \tag{1}$$

在公式中，$\mathcal{T}_{A\leftarrow B}$ 表示图像到变量的映射。在实现中，$\mathcal{T}_{S\leftarrow R}$ 和 $\mathcal{T}_{D\leftarrow R}$ 分别通过 S 和 D 中的关键点检测得到，支持从源和生成的基于像素的文本 gif 中提取关键点轨迹。在实现中，在 MGif 数据集上使用了预训练的 FOMM 模型 [44]，该模型在关键点检测方面表现出了良好的性能。根据预训练模型，对每帧提取的特征进行独立估计，关键点个数设为 10。然而，为了进一步增强情感对生成和分析的整合，本文收集并创建了一个包含 77 个情感标记的动图的 Puppy Maltese 数据集 [39]，并使用它来微调模型。由于 FOMM 难以在不同类别物体的运动传递中获得良好的性能，因此我们仅从关键点检测模块中提取中间结果，并根据我们的任务重新设计后续的生成步骤。

### 3.3 关键点对齐

使用局部仿射变换（local affine transformation）将关键点（GIF）的轨迹与控制点（源文字）对齐；仿射变换矩阵集通过相邻帧的每个点的平移变换计算得到，然后使用基于距离加权插值的方法计算矩阵集的全局非线性变换。

$$\begin{bmatrix} C_j^{f+1} \\ 1 \end{bmatrix} = \sum_{i=1}^{N} w_i(C_j) \cdot \begin{bmatrix} C_j^0 \\ 1 \end{bmatrix} \begin{bmatrix} \mathcal{I} & 0 \\ (X_i^f - X_i^1)^T & 1 \end{bmatrix}, \tag{2}$$

$$w_i(C_j) = \frac{1/\left\|C_j^0 - X_i^1\right\|^e}{\sum_i 1/\left\|C_j^0 - X_i^1\right\|^e}. \tag{3}$$

$C_j^f$ 和 $X_i^f$ 分别表示在帧 f 中的控制点 $j$ 和关键点 $i$。控制点在每一帧的位置参照第一帧的关键点计算，以实现全局稳定。$\mathcal{I}$ 是一个二阶单位矩阵。$w_i$ 是控制点相对于关键点的权重函数。

权重根据 $X_i$ 到 $C_j$ 的相对距离 e-th 次幂的倒数衰减，其中 $e$ 控制仿射变换的局部性，即每个仿射变换对目标点的影响程度。控制点在每一帧的位置参照第一帧的关键点计算，以实现全局稳定。

### 3.4 位置优化

为了减轻由于控制点相对位置变化引起的字形不适当变形，本文基于拉普拉斯坐标对控制点的位置进行逐帧优化，该坐标一般利用邻域信息来描述曲面上的相对位置。对于帧 $f$ 上的控制点 $j$，其拉普拉斯坐标 $L_j^f$ 计算为

$$L_j^f = \sum_{k \in N_j} \omega_{jk}^f (C_k^f - C_j^f) = \sum_{k \in N_j} \omega_{jk}^f C_k^f - C_j^f \tag{4}$$

式中，$C_j^f$ 和 $C_k^f$ 分别为控制点 $j$ 和 $k$ 的笛卡尔坐标。在初始控制点集合的基础上，计算出与 $f$ 的变化不变的初始控制点集合的值，即到控制点 $j$ 的欧氏距离最小的 K-nearest 相邻控制点的集合。$\omega_{jk}^f$ 表示控制点 $j$ 的拉普拉斯表示中相邻点 $k$ 的权值

$$\omega_{jk}^f = \frac{1/\left\|C_k^f - C_j^f\right\|^2}{\sum_{k \in N_i} 1/\left\|C_k^f - C_j^f\right\|^2} \tag{5}$$

考虑到离散采样点分布的不均匀性，本文使用上述基于距离的权重来更好地描述详细的位置信息。此外，还设计了如下目标函数 $\mathcal{L}_{total}$ 来优化由帧获得的控制点序列的坐标。

$$\mathcal{L}_{total} = \alpha \cdot \mathcal{L}_{glyph} + \mathcal{L}_{motion}, \alpha \in [0, +\infty),$$

$$\mathcal{L}_{glyph} = \sum_{j=1}^{M} \left\|L_j^f - L_j^0\right\|^e, \mathcal{L}_{motion} = \sum_{j=1}^{M} \left\|C_j^f - C_j^{f\prime}\right\|^e. \tag{6}$$

$L_j^f$ 和 $L_j^0$ 分别表示帧 $f$ 和帧 0 中控制点 $j$ 的拉普拉斯坐标。其中，“$C_j^f$” 和 “$C_j^{f\prime}$” 为优化前后控制点坐标。$\mathcal{L}_{glyph}$ 测量局部形状细节的保留程度，其计算方法为优化后的控制点与初始控制点之间的拉普拉斯坐标距离之和。$\mathcal{L}_{motion}$ 测量保留了多少运动，作为优化前后控制点距离的总和，即最小化编辑距离。这两个损失函数之间的权衡是一个超参数。越大的 $\alpha$，初始图形的细节越多，保留的运动越少。对于范数 $e$，范数越大，局部性越规范，变形越强。使用 $K$-dimensional 树来加速最近邻搜索，其中参数是经验设置的:$K$= 3。当采用逐帧优化时，可以注意到在损失函数中引入全局时间正则化项以提高平滑性和一致性是值得的。

## 4 复现细节

### 4.1 与已有开源代码对比

这篇论文 wakey 是具有源码的，并且能够正常运行；但是这个 wakey 系统存在一个不足就是无法对不是纯白背景的 GIF 进行处理，因此我添加了一个预处理的模块对非纯白背景的 GIF 进行提取主体并添加白色背景，使用了 yolo 模型进行 GIF 中目标主体检测，得到提示框，然后输入给 SAM 模型对图像进行主体分割，得到主体的 mask 图像，最后与源图像进行

与操作将像素值为 0 的部分设置为纯白色。因此从中参考了 yolo 模型和 SAM 模型官方提供使用代码，之后便是自己对上述代码进行整合以及将图像非主体部分设置为白色；从而得到一个可以对 GIF 进行预处理的模块。

### 4.2 实验环境搭建

AnimaText 系统采用前后端分离的方式搭建，后端使用 flask web 服务器，前端使用 node.js; 测试浏览器我选择谷歌浏览器；

### 4.3 界面分析与使用说明

如图2所示，这个系统由三个板块组成，分别是输入板块，纠正板块，精炼板块。在输入板块，用户需要输入文字，以及一张 GIF，同时系统还提供了修改字体样式和颜色的功能；当用户输入完之后，点击箭头便可以提取 GIF 和文字的关键点并在纠正板块中展示，在纠正板块中，用户可以拖拽控制点来纠正摆放位置不正确的控制点；之后用户可以点击箭头，生成最后的文字动画，在精炼板块，用户可以通过拖拽控制点精细的调整文字的形状，同时可以使用两个超参数实现动画文字的运动幅度和文字细节的权衡。

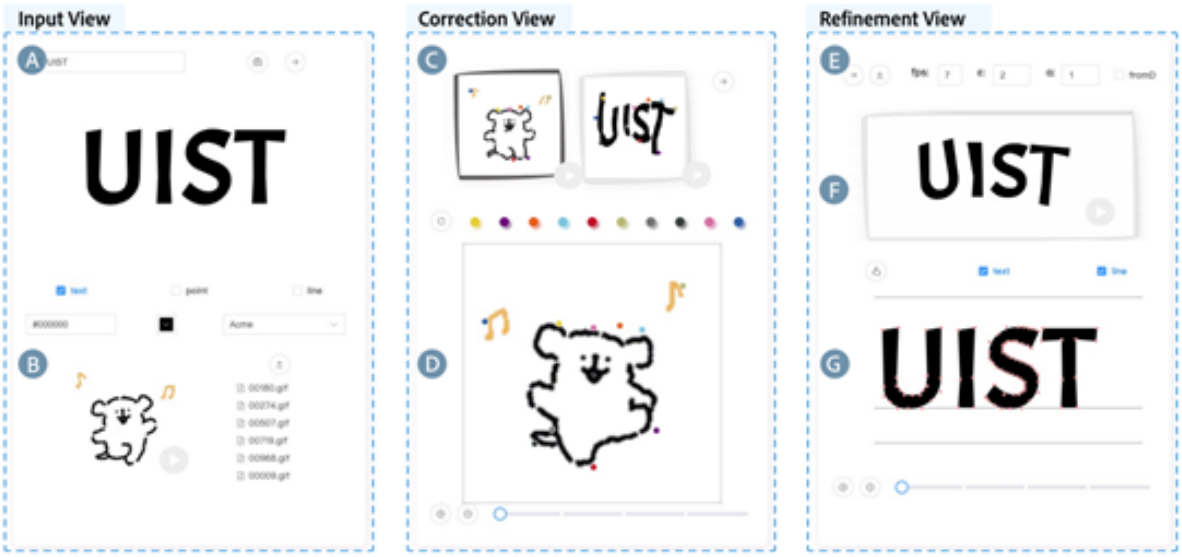


图 2. 操作界面示意图

### 4.4 创新点

本文的对于输入的 GIF 动画具有一个限制，就是必须使用纯白背景动画 GIF；这对于使用者来说是一个很大的限制，因此我的创新点在于，通过使用 SAM 模型 +yolo 模型将复杂背景的 GIF 变为纯白背景的 GIF，从而增加系统的泛化性。效果如下图3。图中左边的 GIF 帧由于背景是黄色的导致图像主体无法被提取控制点，而左边为替换了白色背景的图片，便可以被提取到主体的控制点。

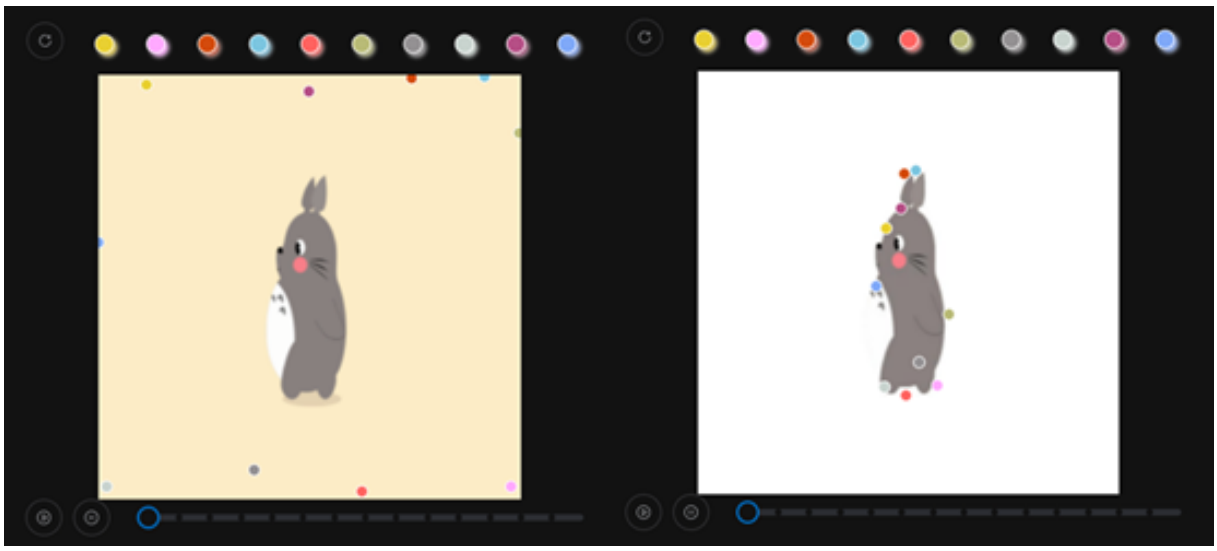

图 3. 实验结果示意图

## 5 实验结果分析

由于缺乏预定义的“基本事实”和缺乏用于定量评估的标准指标，本文通过分析每个组件引入的影响，对本文的方法进行了实证评估。对位置优化组件的评估，实验采用了一个 FOMM 生成的基线与本文提出的方法得到的结果进行对比，结果如图4所示，我们可以看到在引入了位置优化部分时，得到的动画文本细节明显变得更加清晰可见，不会发生文字的裂变。

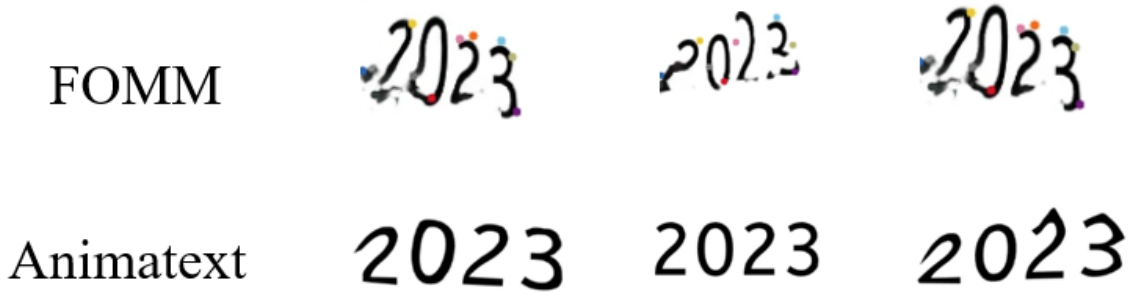


图 4. 效果对比示意图

对于关键点校正组件的评估，模型检测到的关键点$X_i^f$ 可能并不总是准确的，这可能导致依赖这些关键点生成的动画文本发生意外变形。本文的方法允许用户交互式地纠正关键点，从而获得符合他们期望的更理想的生成。如图5所示，通过对前后两帧的分析，我们可以发现，在由 FOMM 生成的基于像素的动画文本图像序列的第四帧中，红色标记的关键点明显向右偏移。这导致在基于这个关键点生成的矢量动画文本中，字母“W”的右下部分过度向右变形。通过将关键点向左拖动到前后帧一致的区域，生成的字形变形显得更加合理和平滑。

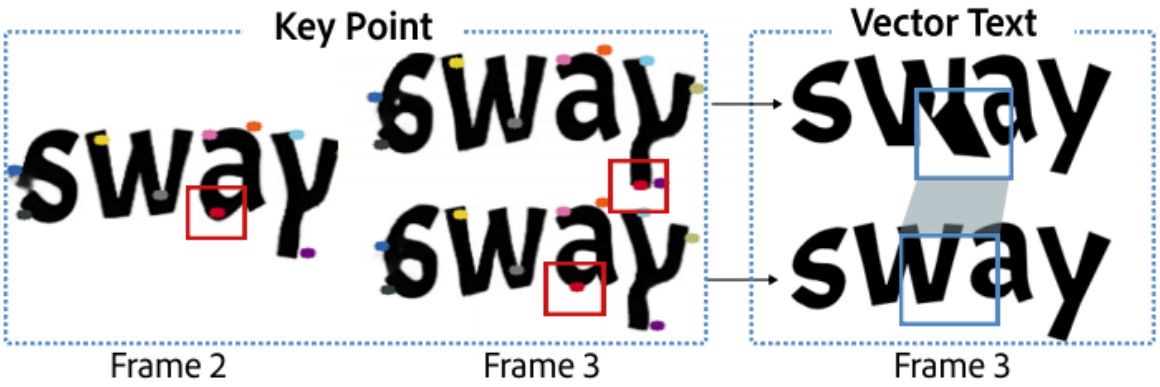


图 5. 效果对比示意图

## 6 总结与展望

本文主要介绍了一个实现将 GIF 动画转移到静态文本上的方法及其原型 wakey-wakey。实现的方法是首先是使用 FOMM 模型对 GIF 和静态文本提取控制点，之后，使用局部仿射变换的方式将静态文本的关键点与 GIF 每一帧的控制点组成的控制点轨迹进行对齐操作，从而生成文本关键点的运动轨迹，然后再进行关键点位置的优化操作，来实现动画文本 GIF 的逐帧文字细节的优化。最后生成文本动画 GIF。此方法由于 FOMM 模型的的限制导致只能针对空白背景的 GIF 进行关键点的提取，虽然我的改进方法可以实现对 GIF 改为白色背景，但是由于 yolo 模型的限制对表情包的动画人物的识别并不好，因此经常会出现无法修改的情况；从而，我想下一步的研究可以从提高框架的泛化性出发，有两个方法实现：第一，对 FOMM 模型进行改进，让其能够对绝大部分的 GIF 提取主体的控制点；第二，根据需要训练相对应的 yolo 模型，比如本文是对表情包的主体检测，因此就可以使用大量的表情包数据去训练一个新的 yolo 模型，来增加模型识别的准确性。

## 参考文献

[1] Adobe inc.2023. after effects.retrieved mar 20, 2023 from. Technical report.

[2] Apple inc.2023. motion. retrieved mar 20, 2023 from. Technical report.

[3] Laurence penny. 1996. a history of truetype. retrieved mar 20, 2023 from https://www.truetype-typography.com.

[4] George Borzyskowski. Animated text: More than meets the eye. In *Beyond the comfort zone: Proceedings of the 21st ASCILITE Conference' , Perth*, pages 141–144, 2004.

[5] Bay-Wei Chang and David Ungar. Animation: from cartoons to the user interface. In *Proceedings of the 6th annual ACM symposium on User interface software and technology*, pages 45–55, 1993.

[6] Marek Dvorožňák, Pierre Bénard, Pascal Barla, Oliver Wang, and Daniel Sỳkora. Example-based expressive animation of 2d rigid bodies. *ACM Transactions on Graphics (TOG)*, 36(4):1–10, 2017.

[7] Shannon Ford, Jodi Forlizzi, and Suguru Ishizaki. Kinetic typography: issues in time-based presentation of text. In *CHI'97 extended abstracts on Human factors in computing systems*, pages 269–270. 1997.

[8] Weston Gaylord, Vivian Hare, and Ashley Ngu. Adding body motion and intonation to instant messaging with animation. In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, New York, NY, USA, 2015. Association for Computing Machinery.

[9] Fa-Ting Hong, Longhao Zhang, Li Shen, and Dan Xu. Depth-aware generative adversarial network for talking head video generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3397–3406, 2022.

[10] Shir Iluz, Yael Vinker, Amir Hertz, Daniel Berio, Daniel Cohen-Or, and Ariel Shamir. Word-as-image for semantic typography. *arXiv preprint arXiv:2303.01818*, 2023.

[11] Daniel G Lee, Deborah I Fels, and John Patrick Udo. Emotive captioning. *Computers in Entertainment (CIE)*, 5(2):11, 2007.

[12] Sabrina Malik, Jonathan Aitken, and Judith Kelly Waalen. Communicating emotion with animated text. *visual communication*, 8(4):469–479, 2009.

[13] Mitsuru Minakuchi and Yutaka Kidawara. Kinetic typography for ambient displays. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 54–57, 2008.

[14] Kyungah Choi Minhwan Kim and Hyeon-Jeong Suk. Yo! enriching emotional quality of single-button messengers through kinetic typography. *In Proceedings of the ACM Conference on Designing Interactive Systems (DIS)*, 2016.

[15] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. *Advances in neural information processing systems*, 32, 2019.

[16] Jiale Tao, Biao Wang, Tiezheng Ge, Yuning Jiang, Wen Li, and Lixin Duan. Motion transformer for unsupervised image animation. In *European Conference on Computer Vision*, pages 702–719. Springer, 2022.

[17] Liwenhan Xie, Xinhuan Shu, Jeon Cheol Su, Yun Wang, Siming Chen, and Huamin Qu. Creating emordle: Animating word cloud for emotion expression. *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[18] Jie Xu and Craig S Kaplan. Calligraphic packing. In *Proceedings of graphics interface 2007*, pages 43–50, 2007.

[19] Yang Zhou, Xintong Han, Eli Shechtman, Jose Echevarria, Evangelos Kalogerakis, and Dingzeyu Li. Makelttalk: speaker-aware talking-head animation. *ACM Transactions On Graphics (TOG)*, 39(6):1–15, 2020.