基于深度学习的二维动画中间帧自动插补方法研究与实现

**1 绪论**

1.1 研究背景及意义

1.1.1 二维动画制作流程

一

1.1.2 二维动画中间帧插补是一个关键且难解决的问题

一

1.1.3 生成式模型在实际制作中仍然存在一些问题

一

1.2 二维动画中间帧插补方法研究现状

1.2.1 基于算法的中间帧插补

（1）插值法-线性插值、贝塞尔曲线插值

（2）关键帧提取法-位移关键帧提取、形状关键帧提取

1.2.2 使用视频帧插值方法进行中间帧插补

一

1.2.3 针对二维动画线稿帧特性的插补方法

一

1.3 生成式模型在二维动画制作中的应用情况

一

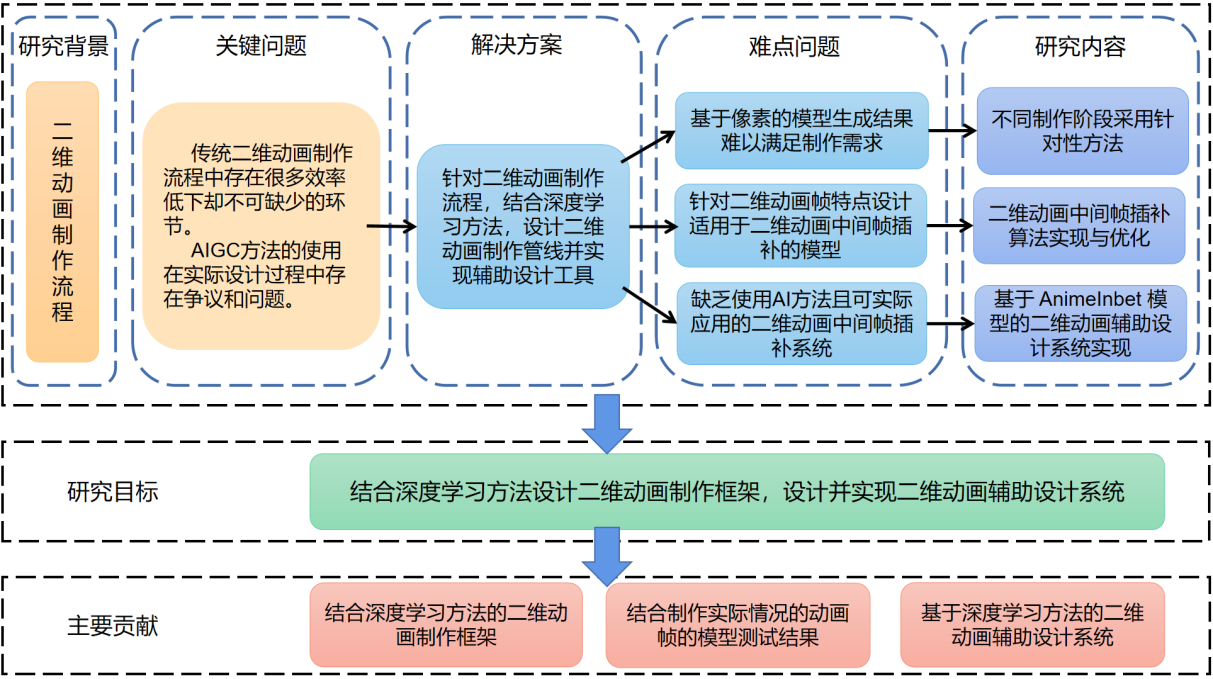
1.4 本文的研究内容和技术路线

1.4.1 研究内容

（研究框架图——解决方案、难点问题——研究内容：三、四、五章标题）

1.4.2 技术路线

一



**2 相关技术介绍**

2.1 动画原理

一

2.2 赛璐珞动画（Cel-animation）

一https://zh.wikipedia.org/wiki/%E4%BC%A0%E7%BB%9F%E5%8A%A8%E7%94%BB

2.3 二维动画中间帧插补技术

2.3.1 基于自动插补算法的二维动画中间帧插补方法

一

2.3.2 基于深度学习的二维动画中间帧插补方法

一

2.4 stable-diffusion-webui介绍

一

2.5 ComfyUI介绍

一

2.6 Qt和PyQt介绍

一

**3 基于深度学习的二维动画制作方案设计**

3.1 二维动画制作的需求分析（用例图）

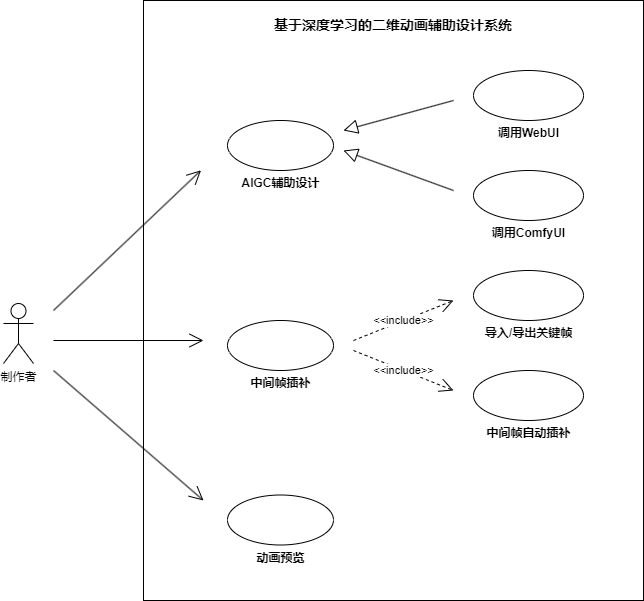


图3.1 基于深度学习的二维动画辅助设计系统用例图

在二维动画的实际制作中，通常将制作流程总结为前期设计、中期制作、后期合成三个环节如图3.2所示（制作流程图）。前期设计包括剧本设计、角色设计、分镜设计等内容，中期制作包括原画绘制、背景绘制、上色、合成等内容，后期合成包括画面编辑、音画合成、视频编辑等内容。在具体的制作中，前期设计与后期合成极大程度依赖制作者的创意、审美等主观因素，对于这一类工作，技术应该实现辅助设计的功能，并且尽可能激发和发挥制作者的创意。中期制作中的很多环节需要大量的重复性工作，对于这些重复性工作，则应该探索提高效率的技术方法。

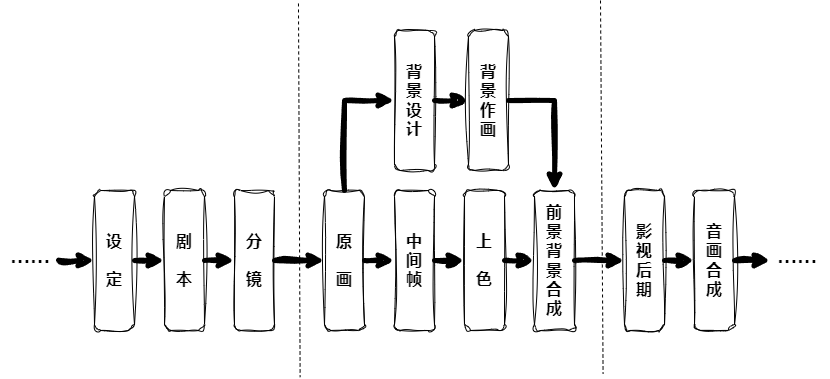


图3.2 二维动画制作流程

根据二维动画制作的实际情况和需求，本文从技术应用的角度将二维动画制作过程简化为三个环节：设计环节（分镜设计）、制作环节（动画帧制作）、预览环节，旨在提高制作效率，探索与前沿研究、最新方法相结合的二维动画辅助设计框架，开发适用于二维动画的实际制作的辅助设计系统。针对三个环节的不同制作需求，可以进行更详细的技术设计。

设计环节主要考虑制作者对分镜设计的需求。分镜是根据剧本等内容制作的体现镜头设计的草图，为后续原画绘制和动画中间帧绘制提供参考和模板，但是由于分镜草图通常比较简单，难以更直观地体现制作者的构思和对画面的预期效果，而AIGC的方法则可以根据草图生成初步的效果预览图，有助于原画师明白导演的意图并在后续环节进行更精确、高效的设计。

制作环节主要考虑动画中间帧插补以及上色的问题和需求。在传统制作中，制作环节主要由原画、中间帧（中割）、上色三个步骤构成，其中前两个步骤通常由画师绘制线稿图，制作好的线稿动画帧在上色环节进行统一填色。原画是动画师根据分镜需求绘制的动画关键帧，原画相当于动画的骨架，而中间帧则作为骨架的填充保证动画的流畅度和连续性，因此中间帧的线条设计与原画基本是相似的（原画和中间帧插图，引用书籍中的图片怎么标注？），但是由于二维动画帧的独特性，导致中间帧插补一直是一个难以彻底自动化的工作，在实际制作时中间帧通常还是由画师亲手绘制，高效且操作方便的中间帧插补方法仍然有待研究。现有的中间帧自动插补方法可以分为基于算法的方法和基于深度学习的方法。基于算法的方法通过计算边点对应关系实现中间帧插补，尽管最新的方法（CACANi）可以实现比较稳定的插补效果，但是在实际操作时需要用户手动确定所有边点对应关系，对于比较复杂的动画帧而言，这种方法会带来额外的工作量。基于深度学习的方法为本文着重探究的方法，通过深度学习框架可以对指定动画帧自动生成中间帧，这极大简化了用户操作，尽管现有的深度学习方法仍然存在一些问题，但这仍然是一个值得深入研究的方向。绘制完毕的线稿动画帧会进入上色环节进行统一填色，在数字绘画技术的辅助下，目前填色已经可以通过绘画软件通过选区填色来完成，但是当动画帧数量庞大时，这仍然会带来不小的工作量，由于相同镜头下的动画帧画面通常十分相似，这意味着相邻帧之间的填色区域通常是高度重合的，于是可以设计最大面积匹配算法来完成上色工作，用户只需要输入同一镜头下的一张已上色的动画帧即可自动对其他相同镜头的动画帧进行一键填色，这能极大提高上色的效率。

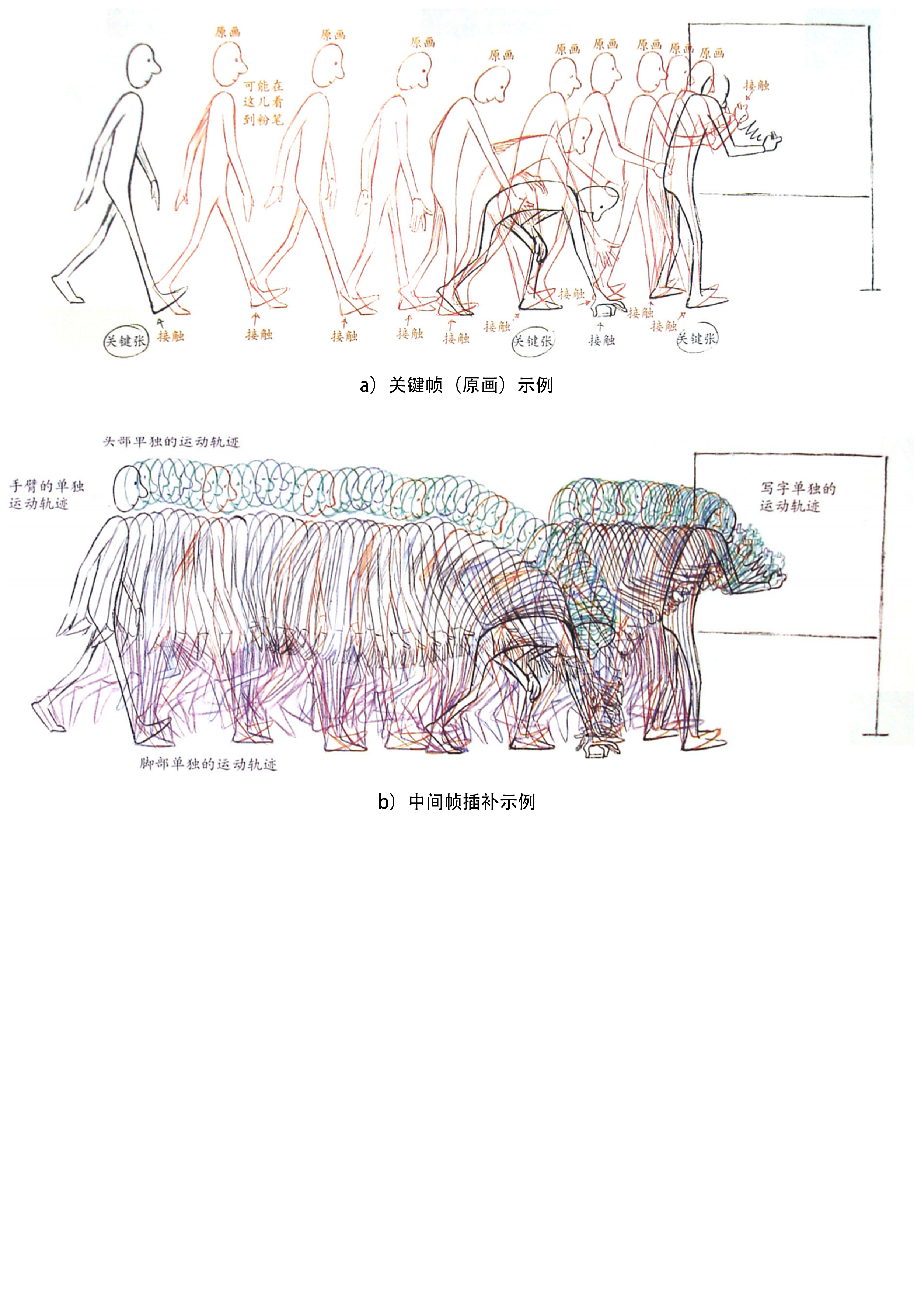


图3.3 关键帧（原画）与中间帧示例图

在本框架中，预览环节为用户提供直观的输出结果展示和动画效果的演示，在实际制作中，制作者还可以在后续环节根据需求对动画添加特效、配音等。

3.2 基于深度学习的二维动画制作方案

3.3.1 基于深度学习的二维动画制作框架

结合3.1节的需求分析，可以设计出如图3.4所示的辅助设计框架图，从需求分析中得到的三个环节入手，该框架旨在针对每个环节的实际制作需求设计相应的解决方案以提高制作效率。

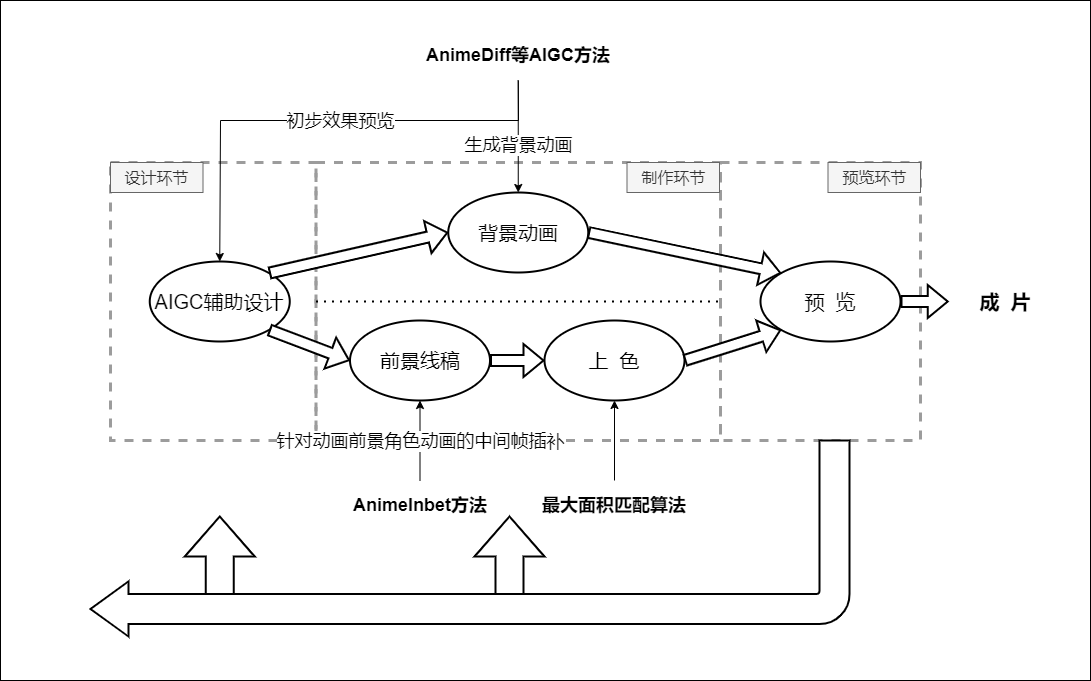


图3.4 基于深度学习的二维动画辅助设计方案

3.3.2 分层制作——前景制作和背景制作的划分

在二维动画的实际制作中，为了丰富画面细节、加强动态效果并提高制作的灵活性，通常会采用多图层设计如图3.5所示（多图层示意图），最常见的情况是将背景和前景分开制作，在必要时甚至会引入更多层次，这种制作方法一方面便于制作者对特定图层进行修改并保证其他图层不受影响，另一方面对于实际制作而言，不同的层次可以应用不同的特效或镜头设计，进而实现更丰富的动态效果，同时对于大型项目而言，分层结构有助于分工合作。

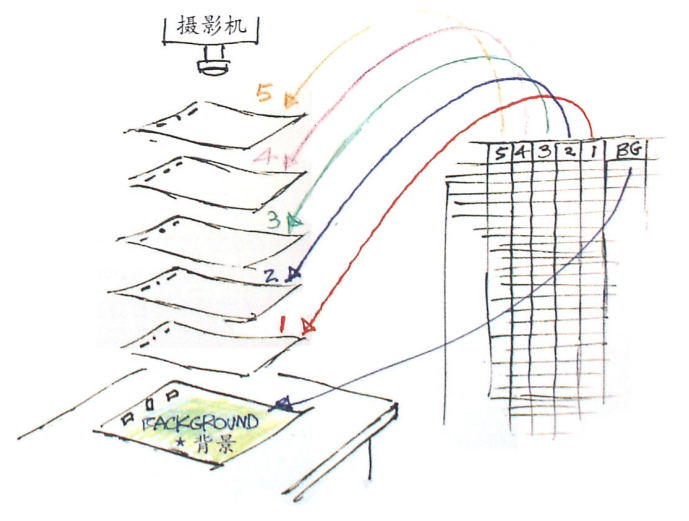


图3.5 多图层示意图

而将深度学习方法应用于二维动画制作仍然存在一些问题，其中比较突出的问题是现有的方法不具备分层生成的能力，只能产生单一图层的结果，单一图层的生成结果在实际制作中不便于修改和微调。通常情况下画师使用数字绘画方法进行画面绘制时采用分图层逐层绘制的方法（分图层绘图和单一图层修改对比图），结合画面的空间结构和透视关系等内容对画面进行分层，这种方法便于画师对画面细节进行微调并不会影响其他图层的内容，尽管少数画师会选择单一图层绘图的方法，但是在实际绘制时仍然采用了分层的思想，鲜有画师在进行创意设计时能不使用任何图层辅助的方法或思想直接平铺出一张图片。而目前常见的深度学习生成图片的方式则更类似后者，深度学习的方法或框架能高效地直接生成单一图层的图像，但是如果从实际制作的角度考虑，这种生成图像的方法实际上十分不利于画师进行微调和修改，而现有的深度学习方法给出的结果通常并不能完全令人满意，当画师对生成结果不满意时只有两种选择——要么重新调整方法重新生成图像，要么在生成的单一图层的图像的基础上进行多图层转绘再修改，从交互效率的角度来看，这两种方法的操作复杂度都比较高。



图3.6 图层对画面修改的影响

尽管深度学习方法在实际应用层面仍然存在不可解决的单一图层问题，这并不意味着这些方法完全不可用。现有的AIGC方法已经能在合理的调试下生成比较高质量的图片，这种方法对于插画等领域带来的影响是十分明显且深远的。由于二维动画对画面流畅度有极高的要求，尽管现有的AIGC方法都或多或少有上下文相关的方法，但是对于实际制作的需求而言，使用AIGC方法直接生成的动画帧之间存在的偏差是不可忽略的，这些细微的、不符合动画原理或角色结构与比例的偏差会让观众轻易地发现动画的不流畅。但是实际制作中也存在一些变化幅度较小的画面，比如动画的背景在一个镜头内通常是静止的或是跟随镜头进行小幅度移动的，对于这类图像的制作AIGC方法能得到比较高质量的结果，且AIGC方法直接生成完成度较高的图像可以省去人工制作时先绘制线稿再上色的流程，在一定程度上提高背景动画制作的效率。

二维动画前景的内容通常是角色的动作，在实际制作这些动画帧时，需要动画师根据动画原理进行动画帧的设计与绘制并进行适当的艺术加工，从而制作出观感流畅、自然的角色动画，而使用现有的AIGC方法直接生成包含角色运动的动画帧时，通常存在相邻帧不连续、生成动画帧不符合运动规律等问题，同时在上文的分析中不难发现通过人工修复这一问题的开销较大。同时，在实际制作中，绘制中间帧的画师需要在原画的基础上通过细微的描改绘制中间帧，而中间帧的数量远远超过原画的数量，这意味着中间帧的绘制是一个繁重、不需要太多设计且无法规避的问题，于是本文在前景制作部分仍然保留动画师绘制原画的工作，着重研究中间帧的自动插补问题的解决方法。

3.3.3 基于AnimateDiff的AIGC辅助设计方法以及背景制作环节的设计

结合制作需求的分析以及方法的调查和研究，本文将AIGC方法应用于二维动画制作中的设计环节和背景动画制作环节。

在设计环节制作者会给出动画的分镜设计草稿，之后动画师会根据分镜草稿、设定图等内容绘制原画、中间帧等内容，分镜草稿决定了后续动画制作的走向，而草稿图一般完成度较低，很难十分详细具体地体现设计细节以及设计者的构思，而现有的AIGC方法支持图生图（img2img）方法，这意味着制作者可以在设计分镜的同时将分镜草稿传入模型生成大致的预览图像，这一功能可以在多部门协同工作时令各制作单位对动画效果有初步的参考，能极大提高沟通合作的效率，图3.7给出了一个简单的利用草图生成参考图的例子（图生图示例）。

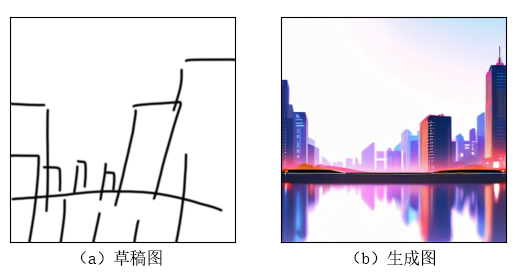


图3.7 图生图（img2img）示例

在制作环节，上文简单分析了AIGC在动画帧制作时仍然存在的弊端，即单一图层不容易进行精确修改、生成相邻动画帧存在“跳变”的现象极大影响动画流畅性和连续性、生成动画帧容易出现不符合动画原理或人体结构等原则的违和画面等问题，但是相比作为主体的前景角色动画，在同一镜头内背景动画通常由静止的一个场景画面构成，在一些情况下可能需要实现镜头移动或缩放的场景动画，而这些简单的动态设计由于不涉及较明显的形变，因而不需要过多考虑画面主体的结构问题，同时对于一些特定的场景的透视关系则可以基于图生图功能输入透视草稿进行规定，因此AIGC高效的生成方法能为场景动画的制作带来极大的效率提升。对于需要镜头移动的背景动画，AnimateDiff提供了一种图生视频的方法，该方法支持单图生成动画以及多图生成动画，对于没有镜头移动的画面可以使用单图生成获得动态的场景画面，对于需要进行镜头移动的画面可以使用多图生成通过输入指定视角的关键帧来生成相应的动画。同时由于AnimateDiff可以直接生成彩色的动画，在调试恰当的情况下可以直接省去背景动画制作中先线稿再上色的流程，进而提高制作效率。

3.3.4 基于AnimeInbet的前景制作环节的设计

早期二维动画制作使用赛璐珞作为胶片，画师在赛璐珞上绘制前景线稿并上色从而实现动画效果，这种作画方式催生出多图层的作画方法，同时这种作画方式使画面由清晰准确的边界线以及大面积色块构成，形成了一种独特的风格——赛璐珞风格，这种动画也被称作赛璐珞动画（wiki）。随着数码绘画的发展，常见的绘图软件都支持多图层绘画，上色则通过对线稿进行选区填色完成。

前景制作主要指角色动画制作，这类动画的制作对动画师的动态造型能力、对角色透视结构的把握、对动画原理的掌握有十分严格的要求，制作优良的角色动画能让角色栩栩如生并体现二维动画的艺术表现力。这一环节的制作可以分为线稿部分和上色部分，动画师根据分镜的需求绘制线稿关键帧（原画），再根据关键帧并结合动画原理描改绘制中间帧，最后进行统一上色。

对于线稿部分，在实际制作中，为了提高动画的流畅性，中间帧的数量远大于关键帧，这导致中间帧绘制一直是一个任务繁重、缺乏创意但不可缺少的工作。在上色阶段，目前主要采用对线稿进行选区填色的方式，这种上色方式要求线稿干净、准确、边界清晰从而保证选区的精准，也正是这一要求导致现有的很多深度学习方法不适用于中间帧插补问题，现有的中间帧插补方法在生成线稿中间帧时会产生边界模糊的情况不利于选区上色，在生成彩色中间帧时由于不能保证生成清晰线条会导致混色等影响观感且不便于修改的问题（图3.8），因此本文在前景制作环节着重探索适合前景动画中间帧的插补方法，尽可能降低用户的操作复杂度且便于后续环节的制作。

对于上色部分，尽管使用选区填色的方法极大提高了动画师的上色效率，但是由于在实际制作时动画帧的数量是庞大的，这导致上色仍然存在不小的工作量。在同一镜头下动画帧中任意区域的颜色通常是固定的，动画师根据事先确定的颜色设定对该镜头下的所有动画帧进行逐一填色。图3.3给出了一段动画的关键帧和中间帧示例，从中不难发现相邻中间帧之间画面的差异较小，相邻动画帧之间对应填色区域之间通常存在较大重合。基于上述思想，可以设计基于最大面积匹配的自动上色算法，动画师只需给定当前镜头下的一帧已上色的动画帧，由算法进行颜色识别以及最大面积匹配对该镜头下的剩余动画帧线稿进行自动上色，该算法的算法流程图如图3.9所示。

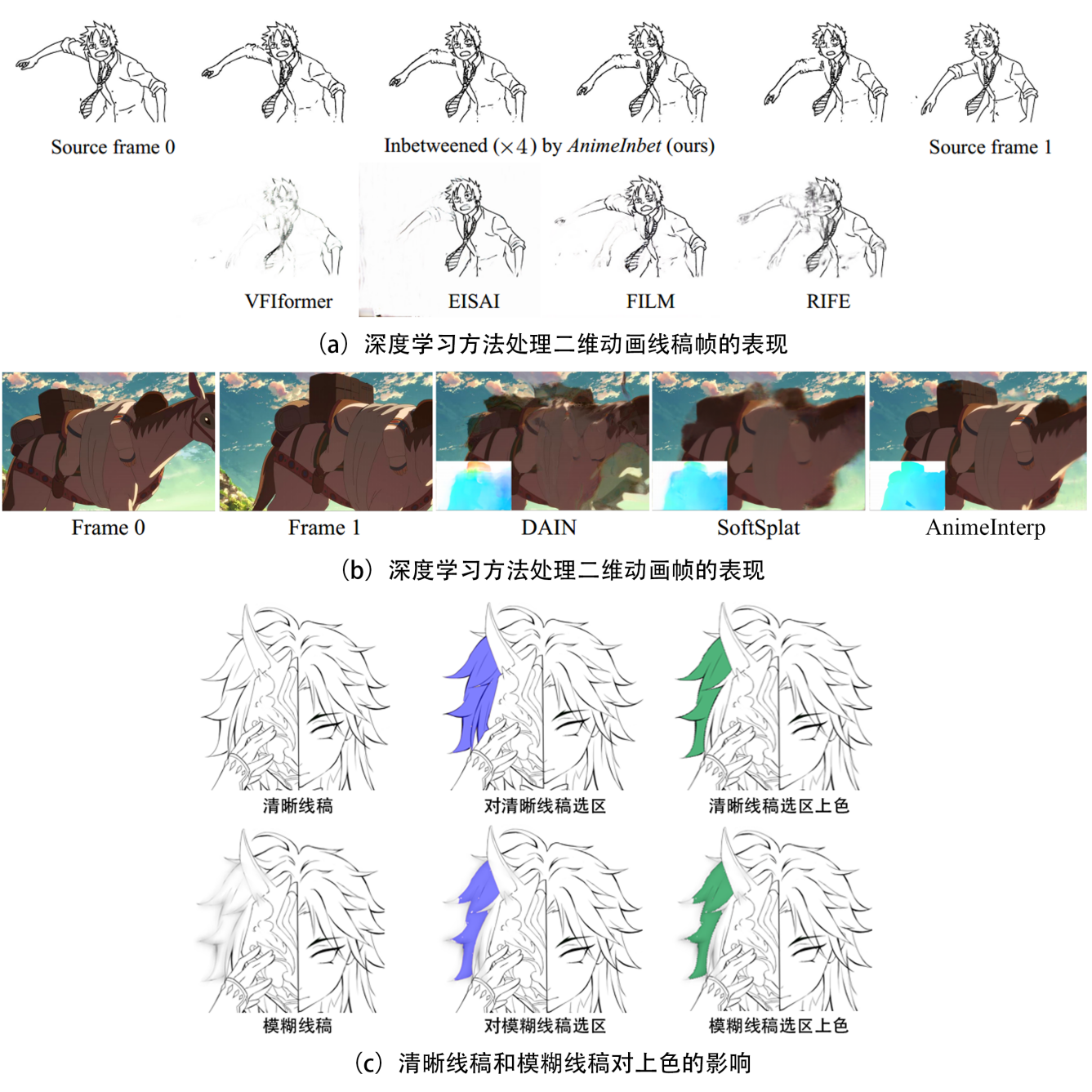


图3.8 深度学习方法导致的线稿模糊、混色问题以及这些问题对上色的影响

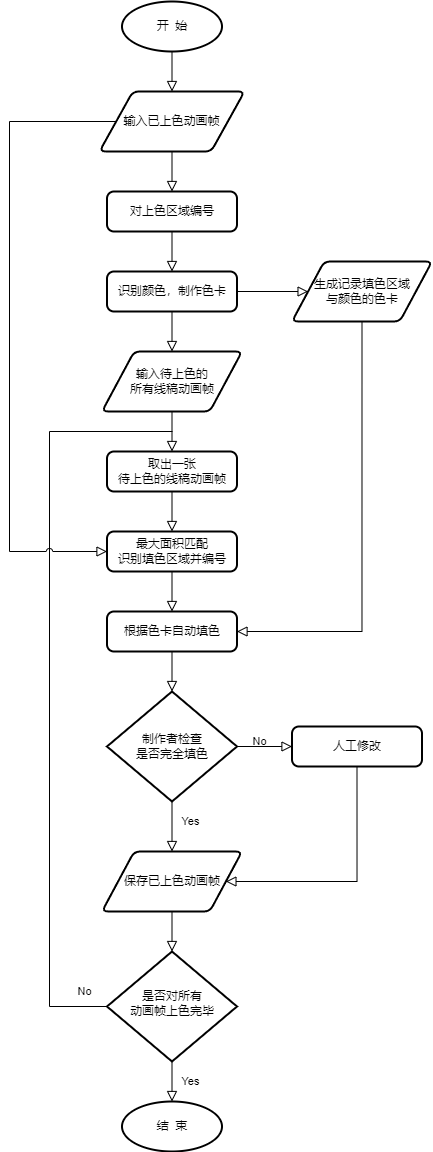


图3.9 最大面积匹配自动上色算法流程图

**4基于深度学习的二维动画制作方法**

4.1 基于AnimateDiff的背景动画生成方法

4.1.1 AnimateDiff的提出背景

近几年来，文生图（T2I）模型由于高质量的生成效果以及文本驱动的可控性而备受研究者、创作者的关注，这种方法为艺术家和业余爱好者等非研究用户进行人工智能辅助内容创作提供了一个低门槛的切入点。随着一些轻量级个性化方法的提出，用户能够使用消费级设备在小型数据集上对这些模型进行定制微调，通过这种方式，用户可以以非常低的成本向预训练的 T2I 模型引入新的概念或风格，从而在CivitAI 和Huggingface等模型共享平台上产生大量由艺术家和业余爱好者贡献的个性化模型。

虽然使用 DreamBooth 或 LoRA 训练的个性化T2I模型由于其杰出的视觉质量吸引了人们的注意，但它们的输出仍然是静态图像，缺乏时间自由度。现有的文本到视频生成方法将时间建模合并到原始T2I模型中，并在视频数据集上调整模型，但是这种方法不利于用户进行个性化设计和调整，因为用户通常无法承担敏感的超参数调整、个性化视频收集和密集的计算资源，这无形中又提高了使用人工智能方法辅助设计的门槛。

对于给定个性化 T2I 模型，例如由用户训练或从 CivitAI或Huggingface下载的模型，在使用较少培训成本甚至无需训练成本的条件下，要将其转变为动画生成器，并保留其原始领域知识和质量，一种简单的方法是通过添加时间感知结构、从大规模视频数据集中学习合理的运动先验来扩充 T2I 模型。然而，对于个性化领域，收集足够的个性化视频来制作视频数据集的成本十分高昂。同时，有限的数据会导致源域知识丢失。

针对上述问题，AnimateDiff单独训练了一个可泛化的运动建模模块，并在推理时将其插入个性化的T2I模型。通过这种方法避免了针对每个个性化模型进行特定调整的工作，并通过保持预训练权重不变来保留它们在原始领域学到的知识。这种方法的另一个关键优势是，一旦模块训练完成，就可以将其插入到任何基于相同基础模型的个性化T2I模型中，而无需进行特定的调整。AnimateDiff的工作流程如图4.1所示，给定一个基本的 T2I 模型，AnimateDiff首先在视频数据集上训练运动建模模块，在此过程中学习运动先验。在训练阶段，仅更新运动模块的参数，从而保留基础 T2I 模型的特征空间，经过这样的训练，对于那些基于 T2I 基础模型训练调整得到的任何个性化模型，运动建模模块都可以将这些模型转变为动画生成器，然后通过迭代降噪生成个性化的动画。

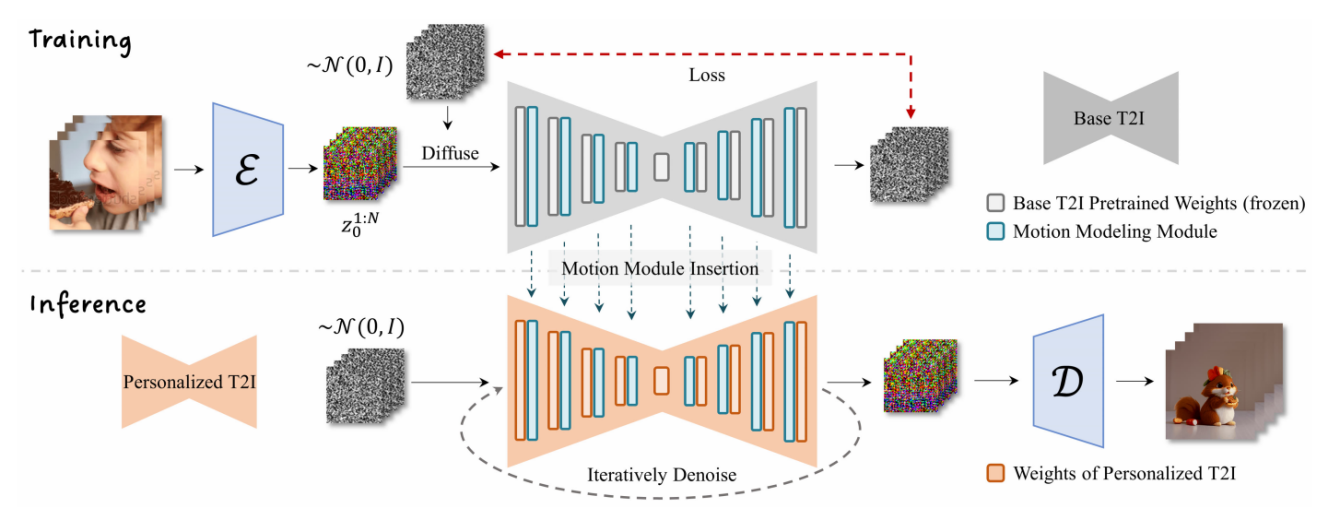
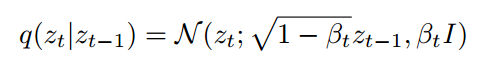


图4.1 AnimateDiff的工作流程

4.1.2 AnimateDiff的实现细节

4.1.2.1 训练阶段通用T2I模型的设计

在训练阶段（Training），为了保证运动建模模块能适用于任何基于基础 T2I 模型得到的个性化模型，需要使用基础 T2I 模型参与运动建模模块的训练。在实际训练中，AnimateDiff选择使用稳定扩散模型（Stable Diffusion，SD） ，SD 基于潜在扩散模型（LDM），它在自动编码器的潜在空间中进行降噪（原文denoise，降噪？去噪？），这种方法能在降低计算成本同时保持高视觉质量。在LDM的训练过程中，输入图像x0首先由冻结编码器（frozen encoder）映射到潜在空间，产生 z0 = E(x0)，再经过预定义的马尔可夫过程进行处理：

（公式）

对于t=1,2,...,T中T是前向扩散过程中的步骤数，超参数βt决定了每一步的噪声强度。上述公式经过迭代后可以整理为：

（公式）

其中。SD的基本训练目标参考了DDPM中提出的：

（公式）

其中 y 是相应的文本描述，τθ(·) 是将字符串映射到向量序列的文本编码器。在SD中，ϵθ(·)使用修改后的UNet来实现，该UNet包含四个下采样/上采样（downsample/upsample）块和一个中间块，从而在网络的潜在空间内产生四个分辨率级别。每个分辨率级别都集成了 2维 卷积层以及自注意力和交叉注意力机制。

4.1.2.2 推理阶段个性化T2I模型的设计

在推理阶段（Inference），DreamBooth和LoRA是两种具有代表性且广泛使用的个性化方法，能让用户根据自己的需求制作个性化的T2I模型。

要获得个性化的T2I模型，一种方法是直接在通用模型的基础上针对特定域的图像进行微调（fine-tuning），但是直接调整模型而不进行正则化通常会导致过拟合（overfitting）或灾难性遗忘（catastrophic forgetting），为了解决这一问题，DreamBooth使用稀有字符串作为指示符来表示目标域，并通过添加原始 T2I 模型生成的图像来扩充数据集，这些正则化图像是在没有指示符的情况下生成的，从而允许模型在微调期间学习将罕见字符串与预期域相关联。

LoRA采取了不同的方法——微调模型权重的残差，即训练 ΔW 而不是 W。微调后的权重计算为 W′ = W + αΔW，其中α是一个超参数，用于控制微调过程的影响，从而为用户提供更多自由来控制生成的结果。为了进一步避免过拟合并降低计算成本，LoRA将 ΔW ∈ R m×n 分解为两个低秩矩阵，即 ΔW = ABT ，其中 A ∈ R m×r ，B ε R n×r ，r ≪ m, n。从训练后的模型参数量（ stores the whole model parameters once trained）来看，LoRA的训练以及用户之间共享的效率比DreamBooth 要高很多。

4.1.2.3 运动建模模块的设计

（1）网络膨胀。由于原始 SD 只能处理图像数据，因此需要模型膨胀使其与运动建模模块兼容，该模块以（批次，通道，帧，高度，宽度）的5维视频张量作为输入，这里采用了类似视频扩散模型的解决方案，通过将帧轴重塑为批次轴，并允许网络独立处理每一帧，将原始模型中的每个2维卷积和注意层转换为仅限空间的伪 3D 层，新插入的运动模块在每个批次的帧之间运行，以实现动画短片中的运动平滑度和内容一致性。细节如图4.2所示。

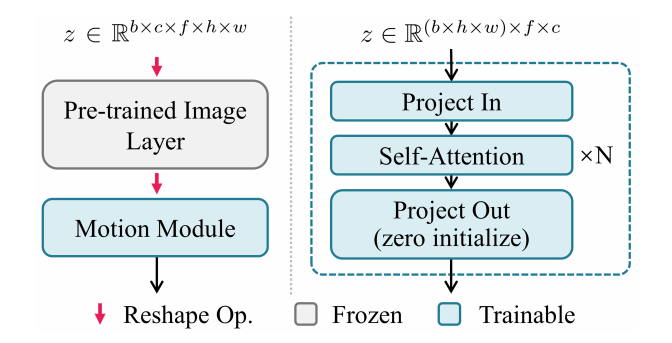
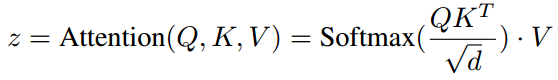


图4.2 运动建模模块设计细节

1. 模块设计。为了实现帧之间有效的信息交换，AnimateDiff基于基本时间变换器（vanilla temporal transformers）设计运动建模模块。基本时间变换器由多个沿时间轴运行的自注意块组成（图4.2）。当通过运动模块时，特征图 z 空间维度的高度和宽度先被重塑（reshape）为批次维度（batch dimension），从而产生帧长的（批次，高度，宽度）序列。重塑后的特征图将被投影并经过几个自注意块，即：

（公式）

其中 Q = WQz、K = W Kz 和 V = WV z 是重构后的特征图的三个投影，此操作使模块能够捕获时间轴上同一位置的特征之间的时间相关性。为了扩大运动模块的感受野，AnimateDiff将其插入到 U 形扩散网络的每个层次，同时还将正弦位置编码添加到自注意力块中，使网络能感知当前帧在动画短片中的时间位置。

（3）训练目标。运动建模模块的训练过程类似于潜在扩散模型（Latent Diffusion Model）。视频数据 x首先通过预训练的自动编码器逐帧编码为潜空间数据（latent data）z。然后，使用定义的前向扩散方法对潜空间数据进行噪声处理：（公式）。将噪声处理后的潜空间数据以及相应提示文本作为使用运动模块膨胀后的扩散网络的输入，使用L2作为损失函数，预测添加到潜空间数据的噪声强度。运动建模模块的最终训练目标是：

（公式）

在训练过程中，基本 T2I 模型的预训练权重被冻结以保持其特征空间不变。

4.1.3 搭载AnimateDiff的ComfyUI

ComfyUI是基于Web开发的开源的Stable Diffusion界面，针对用户个性化工作流程进行了优化并具有更高的灵活性（https://comfyuiweb.com/）。ComfyUI将SD中可控的流程封装成独立的节点，允许用户自行导入模型并通过操作节点自定义工作流生成图片，整个流程不需要更改底层代码，极大降低了使用SD的门槛并让设计流程更加可视化。

4.2 基于AnimeInbet的前景动画辅助设计方法

4.2.1 AnimeInbet的提出背景

卡通动画自 1900 年代初诞生以来，经历了重大变革，早期动画帧是完全由动画师手工绘制的。尽管现在已经有了一些自动化技术来辅助动画制作过程中的一些特定环节，但核心元素——角色线稿——仍然高度依赖手绘，这使得二维动画制作成为劳动密集型产业。开发一种自动化算法，可以从两个输入关键帧生成中间线稿图（通常称为“中间帧”），有可能显着提高生产力。

早期一些基于规则的方法已经被用于严格条件下的中间帧制作，但这些方法通常需要用户亲自指定前后帧线条的对应关系，在实际使用时这种方法需要繁杂的操作，并没有彻底提高用户的操作效率，同时由于严格的使用条件，这种方法在很多常见的应用场景中十分受限。线稿中间帧并不是一般帧插值的一个简单子集，因为线稿的结构极其稀疏。与全纹理图像不同，线稿仅包含大约3%的黑色像素，图像的其余部分是白色背景，这对现有的基于光栅图像的帧插值方法提出了两个重大挑战：（1）由于线稿中缺乏纹理，因此在帧插值中精确计算像素对应关系具有挑战性。一个像素可能有很多相似的匹配候选像素，从而导致运动预测不准确。（2）帧插值方法中常用的扭曲、混合技术会模糊线条和背景之间的明显边界，导致大量细节丢失。尽管目前已经有很多视频帧插值的方法在自然视频以及动画视频上取得了不错的效果，但它们基于像素处理和计算的特点不适合在简洁和稀疏的线稿之间进行中间帧插入，这些方法会产生严重的线条模糊，因此不适合动画的制作。

AnimeInbet给出了一种基于深度学习的解决二维动画中间帧插补问题的方法，同时为了规避基于像素的方法在中间帧插补问题上的局限性，将光栅线条图几何化为矢量图，并将中间任务重新构建为对顶点重新定位的图融合问题，这种方法可以有效地捕捉线条图的稀疏性和独特的结构，同时保留中间帧的细节，基于矢量图的方法得到的插补结果也更加接近实际制作的需求。

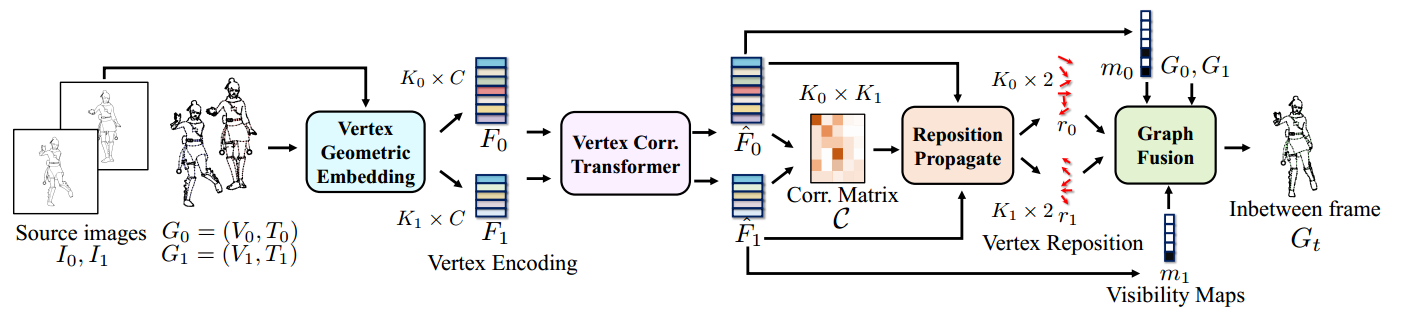


图4.3 AnimeInbet的工作流程

4.2.2 AnimeInbet的实现细节

4.2.2.1 数据预处理

目前艺术创作过程已基本实现数字化，绘图软件可以直接输出矢量格式。不过，对于以光栅格式存储的线稿，可使用一些开源软件或方法来将光栅图转换为所需的矢量格式。这一部分不作为本方法的实现重点也不纳入性能测试。

4.2.2.2 顶点几何嵌入（vertex geometric embedding）

为了实现准确的图匹配，首先需要从矢量图提取顶点特征。线稿图与一般的图不同，除了连线的拓扑结构外，端点的空间位置也决定了线条的几何形状。因此，线稿的几何图形嵌入设计可以分为三个部分：图像上下文嵌入（image contextual embedding）、位置嵌入（positional embedding）、拓扑嵌入（topological embedding），如图4.4所示。

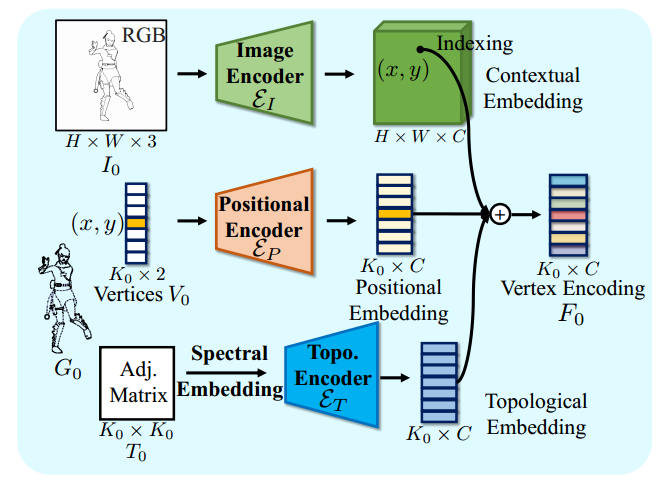


图4.4 顶点几何嵌入的结构设计

（1）图像上下文嵌入（image contextual embedding）：使用二维CNN从输入的栅格图像I中提取深层上下文特征。然后，对于每个顶点，将特征索引为第i个顶点的图像嵌入。

（2）位置嵌入（positional embedding）：使用一维 CNN将每个顶点坐标 (x, y) 映射到 C 维特征。

（3）拓扑嵌入（topological embedding）：为了将拓扑信息纳入低维的特征中，首先对二元邻接矩阵 T 进行频谱嵌入 S，这涉及对图的拉普拉斯矩阵进行特征向量分解，然后将频谱嵌入结果输入后续的一维CNN。

最终的几何图嵌入公式为

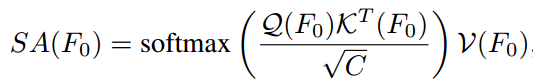
（公式）

F1使用同样的方法获得。

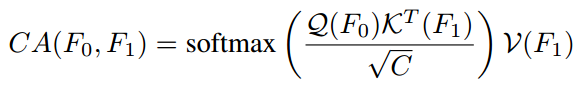
4.2.2.3 顶点对应变换器（vertex correspondence Transformer）

AnimeInbet使用几何特征 F0 和 F1 来建立 G0 和 G1 之间的顶点对应关系，通过计算顶点特征之间的相关矩阵，并将矩阵行列中均有最高匹配度的顶点识别为匹配对。在此步骤之前，首先用一个 Transformer 来聚合图内和图间的相互一致性。

（1）相互聚合（Mutual Aggregation）：首先采用级联交替的自注意力层和交叉注意力层来聚合顶点特征，在自注意力层中，所有查询、键和值都源自单一源特征：

（公式）

其中 Q、K 和 V 分别表示查询、键和值的 MLP。在交叉注意力层中，键和值根据另一个特征计算：

（公式）

如图4.5所示，经过 N 层自注意力层和交叉注意力层后，可以得到聚合特征和。在聚合过程中，每个顶点都代表了同一图内、两个图间所有其他顶点的注意力集合，从而实现了具有相互依赖性的信息全面融合。

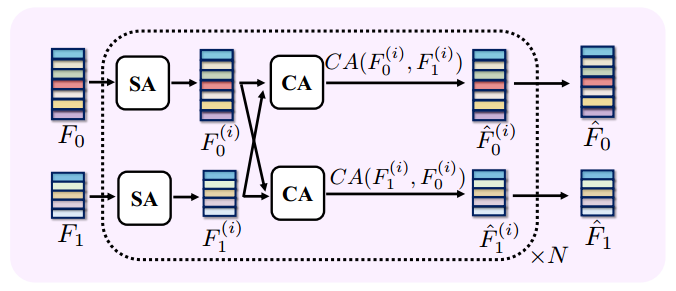
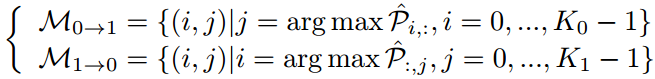


图4.5 顶点对应变换器的结构设计

（3）相关矩阵和顶点匹配（Correlation Matrix and Vertex Matching）：相关矩阵P的计算公式为，之后使用可微最优传输（OT）来提高双重选择一致性并获得，然后以跨行和跨列的arg max值预测从G0到G1的单向匹配，反之亦然：

（公式）

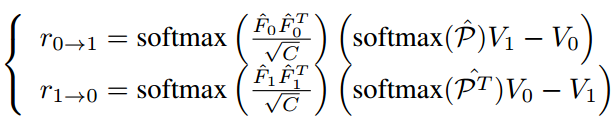
如果顶点对相互一致且相关度大于θ，则被作为最终的对应关系：

（公式）

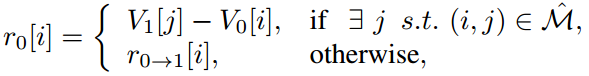
否则，顶点将被忽略。

4.2.2.4 重定位传播（repositioning propagation）

顶点对应中的融合顶点可以根据时间 t 线性地重新定位到中间图 Gt 中的。为了重新定位这些顶点，AnimeInbet设计了一种基于注意力的方案，分别预测和的双向移动向量和：

（公式）

并使用如下公式计算重定位向量：

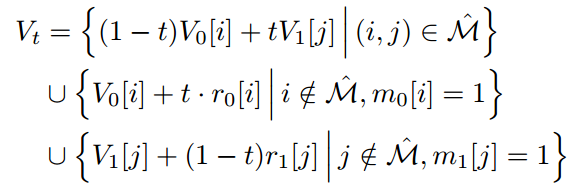
（公式）

r1的计算方法类似。

在这一步中，未匹配顶点的运动向量被计算为中所有顶点移动的softmax均值，即，然后根据得出的自相似性，通过匹配顶点的注意力池对其进行细化，之后，顶点会在新的矢量图中被合理地重定位。

4.3.3.5 目标图合成（graph fusion.）

为了处理线稿中间帧中的遮挡问题，AnimeInbet使用三层 MLP 来预测输入图形的二元可见度图和，即和。然后根据以下方法将两个图中的顶点合并为：

（公式）

这样便实现了与任意时间兼容的重新定位。对于，如果两个端点向量在中都是可见的，则将合并所有原始连接。形式上，如果 或，则，其中和是原始图和合并图中的顶点索引。

4.2.3 最大面积匹配上色算法的设计与伪代码

色卡生成

面积匹配

**5 基于 AnimeInbet 模型的二维动画辅助设计系统实现**

5.1 需求分析与系统设计（用例 功能模块）

一

5.2 基于AIGC的辅助设计模块设计与实现

一

5.3 中间帧插入模块设计与实现

一

5.4 动画预览模块设计与实现

一

5.5 系统测试

一