Journal of System Simulation

Volume 27 | Issue 4 Article 20

8-20-2020

Graphics and Image Local Deformation Based on Moving Least Squares Method

Xiaorong Du
1. School of Physics and Engineering, Sun Yat-Sen University, Guangzhou 510275, China;;

Shuwen Ping
1. School of Physics and Engineering, Sun Yat-Sen University, Guangzhou 510275, China;;

Zhang Yong
2. School of Information Science and Technology, Sun Yat-Sen University, Guangzhou 510006, China;

Graphics and Image Local Deformation Based on Moving Least Squares Method

Abstract

Abstract: Combined with Moving Least Squares method, a deformation function based on control points and area was defined. A localization processing method for image deformation was proposed which extended to vector graph deformation. After adding control points to control deformation and setting the target deformation zone as control area, control points were controlled to deform image and vector graph, which could produce affine transformation, similarity transformation and rigid transformation results. Experiments show that, this method can inhibit boundary aliasing as well as ensuring graphics and image local deformation result smoothness. By applying it to vector graph deformation, multiple complex deformation effects can also be customized.

Keywords

vector graph, deformation, Moving Least Squares, image, local

第 27 卷第 4 期 系统仿真学报© Vol. 27 No. 4
2015 年 4 月 Journal of System Simulation Apr., 2015

基于移动最小二乘法的图形图像局部变形技术

杜晓荣[1]，平淑文[1]，张永[2]
(1. 中山大学物理科学与工程技术学院，广州 510275；2. 中山大学信息科学与技术学院，广州 510006)

摘要：*结合移动最小二乘方法，定义了一个基于控制点和控制区域的变形函数，提出了一种图像变形的局部化处理技术，并将其扩展为适用于矢量图形变形的方法。*通过添加控制点以控制变形，并将目标变形区域设定为控制区域，移动控制点位置，使图像及矢量图形产生变形，生成仿射变换、相似变换及刚性变换结果。实验表明，该方法在保证图像及矢量图形局部平滑变形的前提下，可抑制边界处的混叠；且将其应用于矢量图形变形，可定制多种复杂的变形效果。

关键词：矢量图形；变形；移动最小二乘法；图像；局部
中图分类号: TP391 文献标识码: A 文章编号: 1004-731X (2015) 04-0816-08

Graphics and Image Local Deformation Based on Moving Least Squares Method

Du Xiaorong[1], *Ping Shuwen*[1], *Zhang Yong*[2]
(1. School of Physics and Engineering, Sun Yat-Sen University, Guangzhou 510275, China;
2. School of Information Science and Technology, Sun Yat-Sen University, Guangzhou 510006, China)

Abstract: *Combined with Moving Least Squares method, a deformation function based on control points and area was defined. A localization processing method for image deformation was proposed which extended to vector graph deformation.* After adding control points to control deformation and setting the target deformation zone as control area, control points were controlled to deform image and vector graph, which could produce affine transformation, similarity transformation and rigid transformation results. Experiments show that, this method can inhibit boundary aliasing as well as ensuring graphics and image local deformation result smoothness. By applying it to vector graph deformation, multiple complex deformation effects can also be customized.

Keywords: vector graph; deformation; Moving Least Squares; image; local

引言

图像变形和矢量图形变形技术是计算机视觉领域非常重要的应用，可以实现一副数字图像或图形到另一幅数字图像或图形的平滑过渡，在广告、电影、动画、医学等领域均有广泛应用。相较于图像变形，矢量图形变形速度更快，且不会产生失真。

图像变形过程中，操作者通过添加点、线等控制句柄，并修改控制句柄位置和方向，使得图像以直观的方式变形。为了实现平滑的变形效果，除了控制句柄所操纵的像素点以外，其他像素点也需要根据特定算法进行恰当变形。这些算法可以利用一些插值方法实现，如 RBF, MLS 等。Schaefer 等[1]提出利用移动最小二乘法(MLS)实现图像变形，通过建立并求解特征点图像变换映射函数，以实现图像变形控制。但是利用该方法无法得到良好的局部区域变形结果，原始图像的轮廓会随着控制句柄的修改而出现一定程度变形，甚至会超出图像边界而消失。

本文在 Schaefer 等人工作的基础上，定义了基

收稿日期：2014-01-05 修回日期：2014-04-11；基金项目：广东省省部产学研重点项目(2008A090400013)；2012 年珠海市科技计划项目(2012D0501990009)；
作者简介：杜晓荣(1958-)，男，安徽人，教授，研究方向为软件开发方法及支撑环境、图形图像技术及应用等；平淑文(1991-)，女，山西人，硕士生，研究方向为图形图像技术及应用；张永(1982-)，男，安徽人，博士生，研究方向为计算机图形图像技术及应用。

于控制点和控制区域的变形函数，提出了一种图像变形的局部化处理技术。通过将目标变形区域设定为控制区域，对 MLS 计算结果进行衰减计算，可在保证局部区域平滑变形的前提下，抑制边界处的混叠；并将其扩展为适用于矢量图形变形的方法，应用于 iOS 平台，以实现手机移动端矢量图形快速变形。

1 国内外研究现状

以往图像变形上的工作主要集中在研究不同的控制方法，主要的方法有基于点或者线段特征的图像变形技术。

Sederberg 和 Parry[2]提出了著名的基于网格的三维技术，后应用于利用二维三次曲线参数化图像的自由形式变形(Free Form Deform)。FFD 通过操纵包含物体的一个空间平行点阵来完成变形，被操纵的空间点阵决定了物体的变形函数，该函数指定了物体每个点的新位置，运行速度快。但 FFD 难以准确按照操作者的意图来完成变形，比如，利用 FFD 难以实现某个具体点从原来位置至目标位置的准确移动。Kho 和 Garland[3]提出了一种可直观控制变形的非规则多边形网格的变形技术。Blanco 和 Oliveira[4]提出了基于特征曲线的快速网格变形技术。

Beier 和 Neely 等[5]在网格变形技术的基础上提出了基于域的图像变形技术，根据原始和目标图像中的特征线段定义特征坐标映射，并利用其余点到特征线段的距离确定对准关系。该方法基于 Shepard 的插值，建立平滑变形。但它的运算复杂度高，随着特征线段增多，运算时间急剧上升。

Bookstein[6]基于薄板样条函数提出了一种简单的变形控制点算法。点变形算法建立在对离散特征点插值的基础上，Igarashi 等[7]提出了基于点的图像变形技术，将输入图像进行三角剖分，求解一个未知数数量等于三角形顶点数之和的线性方程组，最大限度上减少图像变形后的撕裂和扭曲，同时确保图像变形尽可能刚性。Levin 等[8]提出了最小二乘法理论模型；基于该模型，Weng 等[9]提出了非线性最小二乘规划算法(Nonlinear Least Squares Optimization)。Schaefer 等[1]提出了一种基于移动最小二乘法(Moving Least Squares, MLS)的图像变形方法, 该方法通过建立特征点、特征线段的图像变换映射函数, 经优化求解，可分别实现图像的仿射变换、相似变换和刚性变换；但是利用该方法无法得到良好的局部区域变形结果，原始图像的轮廓会随着控制句柄的修改而出现一定程度变形，甚至会超出图像边界而消失。本文在 Schaefer 等工作的基础上，提出了一种图像变形的局部化处理技术，可在保证局部区域平滑变形的前提下，有效抑制边界处的混叠。

2 基于 MLS 的图形图像局部变形

2.1 基于 MLS 的图像局部变形

图像变形过程可以看作是在变形函数 f 作用下由原始图像映射到目标图像的过程，将 f 作用于原始图像上的每一个像素点 v，就可以得到变形后的图像。基于 MLS 的图像局部变形方法首先利用基于 MLS 的变形函数计算像素点映射位置，然后对映射结果进行插值处理，从而实现图像局部变形。

2.1.1 MLS 变形

假设 p 为原始控制点的集合，q 为经过拖动后控制点的集合，f 为变形函数，根据 Levin 等[8]提出的移动最小二乘法(MLS)理论模型，对于图像中任意像素点 v，f 应使下式取最小值：

$$E=\sum_i w_i\left|f(p_i)-q_i\right|^2 \tag{1}$$

其中 p_i 和 q_i 是控制点集中的点的坐标，用行向量表示；权重 w_i 为 $w_i=\dfrac{1}{\left|p_i-v\right|^{2\alpha}}$，其中 α 为调节变形效果的参数，本文取 1。由于 w_i 取值随着 v 不同而变化，故该方法称为移动最小二乘法。当 v 等于 p_i 时， w_i 趋近于无穷，f 函数为： $f\left(p_i\right)=q_i$；如果对于每一个控制点 $p_i=q_i$，则 $f\left(v\right)=v$。

本文主要讨论仿射变换情形，假设变形函数 f 包含线性变换项 M 和平移变换项 T 两部分，则有：

$$f(x)=Mx+T \tag{2}$$

第 27 卷第 4 期 系统仿真学报 Vol. 27 No. 4
2015 年 4 月 Journal of System Simulation Apr., 2015

根据最小化原理，简化公式，可得：

$$f(x)=(x-p_*)M+q_* \tag{3}$$

其中，q_*和 p_*为加权质心：

$$p_*=\frac{\sum_i w_i p_i}{\sum_i w_i} \qquad q_*=\frac{\sum_i w_i q_i}{\sum_i w_i}$$

将(3)代入(1)，有：

$$E=\sum_i w_i\left|\hat{p}_i M-\hat{q}_i\right|^2 \tag{4}$$

其中：

$$\hat{p}_i=p_i-p_* \qquad \hat{q}_i=q_i-q_*$$

上述矩阵 M 可以视为一般的仿射变换，包含了缩放、错切、旋转等变换成分。通过分析得：

$$M=\left(\sum_i \hat{p}_i^T w_i \hat{p}_i\right)^{-1}\sum_j w_j \hat{p}_j^T \hat{q}_j$$

利用 M 的闭合解，f 函数可写作：

$$f(v)=(v-p_*)\left(\sum_i \hat{p}_i^T w_i \hat{p}_i\right)^{-1}\sum_j w_j \hat{p}_j^T \hat{q}_j+q_* \tag{5}$$

用户在拖动控制点进行变形交互的过程中，p 固定不变，故(5)式可以写作：

$$f(v)=\sum_j A_j \hat{q}_j+q_* \tag{6}$$

其中，A_j是一个标量，由 v 和 p 决定，计算公式如(7)所示。通过预计算 A_j 值，可加快变形计算速度。

$$A_j=(v-p_*)\left(\sum_i \hat{p}_i^T w_i \hat{p}_i\right)^{-1} w_j \hat{p}_j^T \tag{7}$$

图 1 所示为 MLS 图像变形效果示例。图中(a)为卡通角色原始图像，通过在图像中适当位置处添加控制点，并将其移动至新位置，对图像进行变形；(b)为利用仿射变换变形函数计算生成的变形结果，图像中卡通角色具有非均匀缩放。本文所有变形效果示例中，均用红色圆点表示各控制点初始位置，绿色圆点表示各控制点被拖动后的目标位置。

(a) 原始图像 (b) 仿射变换

图 1 MLS 图像变形效果示例

还可以根据(3)式分析计算得出相似变换及刚性变换情况的变形函数，计算结果见附录，具体细节参见文献[1]。

2.1.2 基于 MLS 变形的局部化处理技术

利用上述变形方法并无法获得良好的图像局部变形结果。如图 2 所示，随着控制点位置的改变，原始图像的轮廓会有一定程度的变形；在控制点集中分布在局部区域时，控制点移动较小位移，就会出现轮廓超出图像边界的情况。

(a) 原始图像 (b) 仿射变换

图 2 MLS 仿射变换结果轮廓丢失

为解决这一问题，我们对 MLS 变形映射结果进行了后期插值处理，提出了图像变形的局部化处理技术；在控制点控制变形的基础上，通过设定控制区域，可在保证局部平滑变形的前提下，抑制图像边界处的混叠。

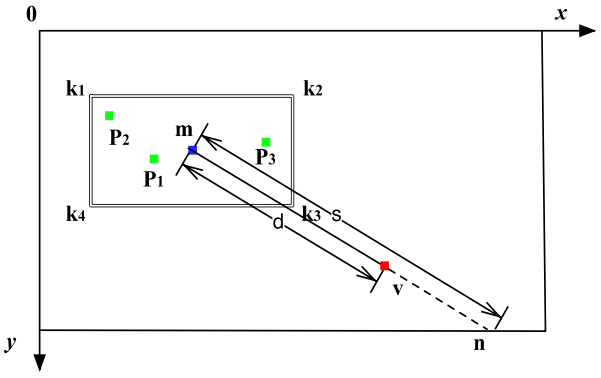


图 3 基于控制点和控制区域的 MLS 变形计算示意图

如图 3 所示，以局部矩形区域变形为例，实线为图像边界，将目标变形区域设定为控制区域，边界以双描线表示，v 是图像上任意一像素点，p_1，p_2，p_3……为控制点，k_1，k_2，k_3……分别表示控制区域顶点，m 点为控制区域的几何中心点，表达式如下：

$$m=\frac{\sum_{i=0}^{x-1}k_i}{x}$$ (x 为控制区域顶点数)

从 m 点向 v 点绘制一条线段，延长线交图像边界于 n 点，mv 长度记为 d，mn 长度记为 s。定义 $l=\frac{d}{s}$，$l\in[0,1]$，当 m、v 重合时，l 取值为 0；当 n、v 重合时，l 取值为 1。以 l 为自变量，定义一个高斯函数 $g(l)$，表达式如下：

$$g(l)=e^{-\frac{l^2}{\sigma^2}}$$

其中，σ 为影响系数，取值范围为 $(0,\infty)$，取值越大，拖动控制点对控制区域以外的图像像素点影响越大。在图像及控制区域确定时，对于每一个像素点 v, d, s 有不同值，故，l 及 $g(l)$ 可写作：

$$l(v)=\frac{d(v)}{s(v)}$$

$$g(v)=e^{-\frac{l^2(v)}{\sigma^2}}=e^{-\frac{d^2(v)}{s^2(v)\sigma^2}} \quad (8)$$

此外，定义一个门函数 $G_A(v)$，表达式如下：

$$G_A(v)=\begin{cases}1 & v\text{在控制区域内}\\ 0 & v\text{不在控制区域内}\end{cases} \quad (9)$$

据此定义新的图像变形函数 $f'(v)$，像素点 v 的变形函数修改为：

$$f'(v)=(f(v)-v)\max(g(v),G_A(v))+v \quad (10)$$

其中 $f(v)$ 为附录中仿射变换、相似变换或刚性变换情况的变形函数，$\max(g(v),G_A(v))$ 表示取 $g(v)$ 和 $G_A(v)$ 中较大值。利用变形函数 $f'(v)$，计算图像上每个像素点变形后的新位置，将该像素点的 RGBA 信息依次复制到目标图像对应位置上，即可生成变形后的图像。

目前，图像变形映射方式主要有两种：正向映射和逆向映射。考虑到正向映射重建的目标图像有“空洞”和“重叠”等缺陷，本文采用逆向映射，从目标图像出发，利用变形函数 $f'(v)$ 求解目标图像上每个像素点在原始图像上的对应位置，需注意，使用逆向映射无法预计算 A_j 值。当图片较大时，映射计算比较耗时，故本文不遍历目标图像每个像素点，而是将目标图像切割为长宽为 $w\times h$ 的多个矩形，计算每个矩形的四个顶点在原始图像上的对应位置，然后在矩形内利用双线性插值填充。W, h 取值决定了计算速度及变形精度，取值越大，计算速度越快；取值越小，变形精度越高、变形效果越好[10-12]。

图像局部变形主要步骤如下：

(1) 计算控制区域几何中心点 m 的位置行向量；

(2) 将目标图像切割为长宽为 $w\times h$ 的多个矩形；

(3) 利用(10)式及 m 值，计算每个矩形的四个顶点在原始图像上的对应位置；

(4) 根据原始图像与目标图像的对应关系，利用双线性插值填充目标图像上各矩形，即可得到变形结果。

以仿射变换为例，图 4 所示，为应用上述算法及映射方式，计算得到的变形结果示例。图 4(a)为原始图像，图 4(b)为利用原始仿射变换算法得到的变形图像，图 4(c)为利用修改后的局部仿射变换算法得到的变形图像，矩形框为控制区域边界线。图 4(b)及图 4(c)中，控制点数目及各控制点原始位置、拖动后位置均一致，可以看到，图 4(b)中，角色轮廓超出了图像边界，而图 4(c)中，只是局部放大了嘴巴所在区域。

(a) 原始图像 (b) 原始仿射变换 (c) 局部仿射变换

图 4 原始仿射变换与局部仿射变换效果比较

2.2 基于 MLS 的矢量图形变形

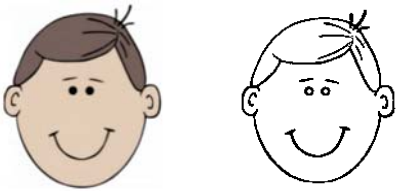
2.2.1 图像矢量化

图像矢量化研究始于二十世纪七十年代，但是尚未有一套通用的、高精度、高速度的算法。本文采用 Canny 算子实现图像矢量化[13]。

首先获取一张图片，并将其灰度化处理；然后对其进行中值滤波，在保存边缘信息的前提下，去

噪；之后利用 Canny 算子提取边缘信息，并利用 8-freeman 链码对二值图像进行编码，提取轮廓线信息[14]；最后将提取到的边缘信息进行拟合，得到矢量化结果。

图 5 所示，为图像矢量化效果示例。图(a)为原始图像，图(b)为经过矢量化后得到的矢量图形。



(a) 原始图像 (b) 矢量图形

图 5 图像矢量化效果示例

2.2.2 基于 MLS 的矢量图形变形

矢量图形变形函数与图像局部变形函数相似，以仿射变换为例，表达式如下：

$$f_{av}(r)=(\sum_j A_j\hat{q}_j+q_*-r)\max(g(r),G_A(r))+r \quad (11)$$

其中，

$$A_j=(r-p_*)\left(\sum_i \hat{p}_i^T w_i \hat{p}_i\right)^{-1} w_j \hat{p}_j^T \quad (12)$$

$$g(r)=e^{-\frac{l^2(r)}{\sigma^2}}=e^{-\frac{d^2(r)}{s^2(r)\sigma^2}}$$

$$G_A(r)=\begin{cases}1 & r\text{在控制区域内}\\ 0 & r\text{不在控制区域内}\end{cases}$$

d, s, p_*, q_*, $\hat{p}_i$, $\hat{q}_i$, max 定义同上，r 为矢量图形任意一特征点。

与图像变形不同，矢量图形变形采用正向映射，变形过程主要步骤如下：

(1) 利用(12)式预计算各位置的 A_j 值，及控制区域几何中心点 m 的位置行向量；

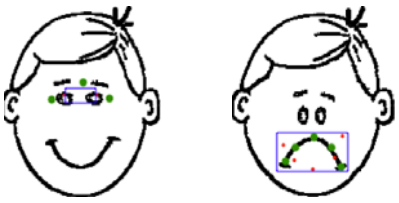
(2) 利用(11)式及已得到的 A_j 值和 m 点位置，逐个计算各特征点变形后的新位置，得到新的特征点集；

(3) 将新的特征点集进行拟合，得到变形后的矢量图形。

因为只需要计算原始图形上各个特征点在目标图形上的新位置，且采用正向映射可预计算 A_j 值，故矢量图形变形计算速度很快。

上述矢量图形变形方法不受图形大小、复杂度等约束，适用于任意矢量图形变形。

图 6 所示，为图 5(b)中的矢量图形变形效果示例。原始图像如图 5(a)所示，原始矢量图形如图 5(b)所示，图 6(a)为眼睛放大效果示例，图 6(b)为嘴巴下弯效果示例。



(a) 眼睛放大 (b) 嘴巴下弯

图 6 矢量图形变形效果示例

3 实验验证

实验所用微机处理器为 2.3GHz Intel Core i7，内存为 4GB，iPhone 模拟器版本为：Xcode 5.0.2，iOS 7.0。

3.1 变形效果及运算时间比较

以仿射变换为例，我们对原始变换方法及局部变换方法进行了比较，图像及矢量图形变形效果示例如图 7 所示。各控制点移动方向及距离如 A 组图片所示；B, C, D, E 分别为图像和矢量图形利用原始仿射变换方法和局部仿射变换方法，对 A 组中对应控制方式进行计算所得到的变形结果。图像变形实验中，将图像切割为 20×20 的正方形进行变形计算。各组变形运算时间如表 1 所示。

表 1 各组变形运算速度比较

变形方法 \ 控制点个数		3 (图 1)	5 (图 2)	8 (图 3)
图像	原始仿射变换 (B 组)	13.7ms	19.5ms	27.4ms
	局部仿射变换 (C 组)	14.4ms	20.4ms	28.3ms
矢量图形	原始仿射变换 (D 组)	6.1ms	9.3ms	14.0ms
	局部仿射变换 (E 组)	7.3ms	10.2ms	15.0ms

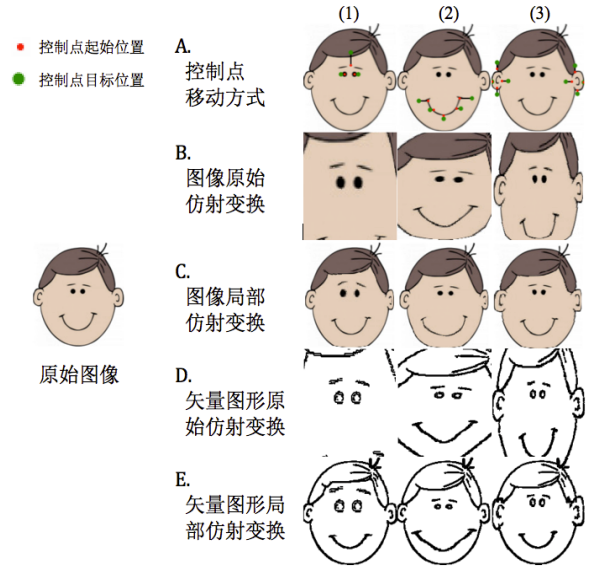


图 7　图像及矢量图形变形效果比较

由表 1 可以看出，两种仿射变换的变形函数运算时间均随控制点个数增加而延长；同等条件下，局部仿射变换相较原始仿射变换方法多耗时约 1ms，故局部仿射变换算法相较原始仿射变换算法并没有造成太大计算负担；此外，示例中原始图像尺寸为 150×168，矢量图形变形运算时间约为图像变形的一半左右，当原始图像尺寸较大时，矢量图形变形速度优势会更明显。

从图 7 可以看出，C, E 组(1)、(2)、(3)分别只对眼睛、嘴巴、耳朵进行了变形，人物脸部及头部轮廓未有明显变形；而 B, D 组图像或图形轮廓均已超出边界。故，局部变换方法可在对局部进行变形的前提下，抑制图像或图形边界处的混叠，变形结果平滑、真实感强。

3.2 矢量图形变形效果

随着 3G 网络的普及，越来越多的软件从大屏幕转战小屏幕，运行在以 iPhone 和 Android 手机为代表的智能机上；而在 App store 及各大安卓 App 平台上，尚未有一款能够进行图像矢量化及矢量图形变形的应用。为此，我们尝试将本文提出的算法及变形公式应用于 iOS 平台，以仿射变换为例，进行矢量图形变形实验和效果分析。

图 8 所示，为 App 界面。使用时，首先根据原始图像获取矢量图形，然后设定控制区域并添加控制点，随后拖动各控制点至目标位置，点击变形按钮即可得到变形结果。

图 9、10 所示，为使用上述 App 实现的矢量图形变形效果示例，为清晰地显示变形结果，每个效果分别用显示控制点和隐藏控制点两张效果图表示。

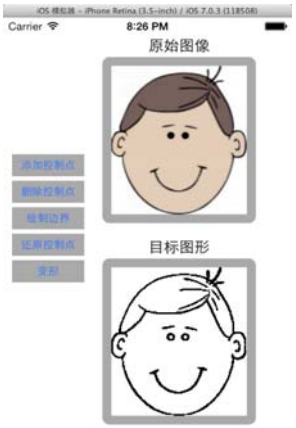


图 8　iOS 平台图像矢量化及矢量图形变形 App

图 9 所示为人物五官变形效果示例，A 组为嘴部变形效果示例，分别表现了委屈、难过、微笑 3 种表情；B 组为眼睛变形效果示例，分别为眼睛放大、向左看、眼睛缩小三种变形效果；C 组为脸型变形效果示例，分别为瓜子脸、国字脸、小脸型 3 种变形效果。可以看出，通过设定控制区域，并添加少量控制点，即可方便地得到难过、忧伤、欣喜、开心等多种表情，以及实现调整人脸胖瘦、眼部动作等多种需求。

图 10 所示为卡通角色动作变形效果示例。图(a)为原图，为提高真实度，矢量化过程中添加了色彩信息；图(b)中，将 m 豆小人右胳膊区域设定为控制区域，并添加多个控制点，通过移动控制点，使得 m 豆小人右胳膊向下弯曲，变形后的 m 豆小人身体、双腿及左胳膊区域没有明显变形，图(c)中，将 m 豆小人双腿区域设定为控制区域，利用控制点将 m 豆小人右腿缩短、左腿延长，以此模仿 m 豆小人奔跑的动作，变形后的 m 豆小人其余部分没有明显变形。

组合图 10 中所示操作，可以实现复杂的卡通

第 27 卷第 4 期 系统仿真学报 Vol. 27 No. 4
2015 年 4 月 Journal of System Simulation Apr., 2015

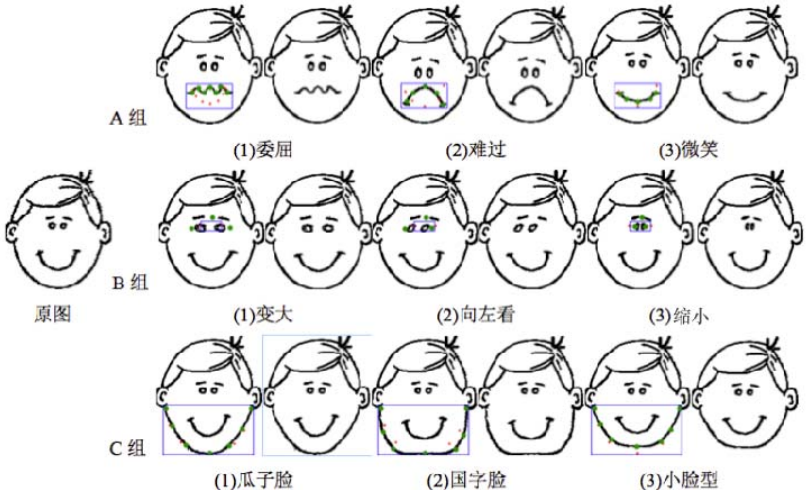


图 9　人物五官变形效果示例

(a) 原图　　(b) 右胳膊弯曲　　(c) 奔跑

图 10　卡通角色动作变形效果示例

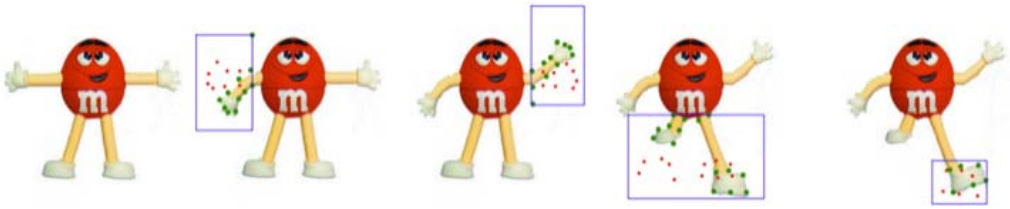


图 11　卡通角色全身动作夸张变形效果示例 1

图 12　卡通角色全身动作夸张变形效果示例 2

角色夸张变形，效果示例如图 11 所示。图 12 所示为另一卡通角色全身动作夸张变形效果示例。综合图 10～12 可以看出，利用矢量图形变形方法可以对卡通角色的全身动作进行夸张变形，并得到良好的变形结果；该方法可应用在广告、游戏和动漫产业等领域，降低设计与制作难度，缩短开发周期。

上述实例表明，利用矢量图形变形方法可以很方便地定制人物表情、动作等，实现复杂的卡通角色全身动作夸张变形，模拟多种场景，变形效果平滑、真实感强，且可抑制图形边界处的混叠。

4　结论

本文在 Schaefer 等人工作的基础上，定义了基于控制点和控制区域的变形函数，提出了一种图像变形的局部化处理技术；并将其扩展为适用于矢量图形变形的方法，成功运用在基于 iOS 平台的矢量

图形变形 App 上，运行效果良好。通过设定控制区域，局部变换方法可在保证局部关键区域平滑变形的前提下，抑制图像及图形边界处的混叠。实例显示，利用局部变换方法，可方便地定制人物表情、动作等，实现复杂的卡通角色夸张变形，模拟多种场景，变形效果平滑、真实感强。今后，结合特征点提取及控制点自动生成技术，上述算法将被应用于卡通角色动作、表情智能定制的研究中。

参考文献：

[1] Schaefer S, McPhail T, Warren J. Image Deformation Using Moving Least Squares [C]// ACM Transactions on Graphics (TOG). USA: ACM, 2006, 25(3): 533-540.

[2] Sederberg T W, Parry S R. Free-form Deformation of Solid Geometric Models [C]// ACM Siggraph Computer Graphics. USA: ACM, 1986, 20(4): 151-160.

[3] Kho Y, Garland M. Sketching Mesh Deformations [C]// ACM SIGGRAPH 2007 Courses. USA: ACM, 2007: 41.

[4] Blanco F R, Oliveira M M. Instant Mesh Deformation [C]// Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games. USA: ACM, 2008: 71-78.

[5] Beier T, Neely S. Feature-based Image Metamorphosis [C]// ACM SIGGRAPH Computer Graphics. USA: ACM, 1992, 26(2): 35-42.

[6] Bookstein F L. Principal Warps: Thin-plate Splines and the Decomposition of Deformations [J]. IEEE Transactions on pattern analysis and machine intelligence (S0162-8828), 1989, 11(6): 567-585.

[7] Igarashi T, Moscovich T, Hughes J F. As-rigid-as-possible Shape Manipulation [C]// ACM Transactions on Graphics (TOG). USA: ACM, 2005, 24(3): 1134-1141.

[8] Levin D. The Approximation Power of Moving Least-squares [J]. Mathematics of Computation of the American Mathematical Society (S0025-5718), 1998, 67(224): 1517-1531.

[9] Weng Y, Xu W, Wu Y, *et al*. 2D Shape Deformation Using Nonlinear Least Squares Optimization [J]. The visual computer (S0178-2789), 2006, 22(9-11): 653-660.

[10] Zhang Y, Lai J, Du X. Vision-Related MLS Image Deformation Using Saliency Map [C]// Image and Graphics (ICIG), 2011 Sixth International Conference on. USA: IEEE, 2011: 193-198.

[11] Weng Y, Shi X, Bao H, *et al*. Sketching MLS Image Deformations on the GPU [C]// Computer Graphics Forum. USA: Blackwell Publishing Ltd, 2008, 27(7): 1789-1796.

[12] 华顺刚, 李晓晓, 李绍帅. 利用控制曲线和移动最小二乘法实现图像变形 [J]. 小型微型计算机系统, 2010, 31(11): 2251-2254.

[13] 张小洪, 杨丹, 刘亚威. 基于 Canny 算子的改进型边缘检测算法 [J]. 计算机工程与应用, 2003, 39(29): 113-115.

[14] 鲁光泉, 许洪国, 李一兵. 基于链码检测的直线段检测方法 [J]. 计算机工程, 2006, 32(14): 1-3, 10.

附录(表 2)：

表 2 仿射、相似、刚性变换情况的变形函数表达式

	M 矩阵表达式	预计算公式
仿射变换	$M=\left(\sum_i \hat{p}_i^T w_i \hat{p}_i\right)^{-1} \sum_j w_j \hat{p}_j^T \hat{q}_j$	$f_a(v)=\sum_j A_j \hat{q}_j + q_*$ $A_j=(v-p_*)\left(\sum_i \hat{p}_i^T w_i \hat{p}_i\right)^{-1} w_j \hat{p}_j^T$
相似变换	$M=\frac{1}{\mu_s}\sum_i w_i \begin{pmatrix} \hat{p}_i \\ -\hat{p}_i^{\perp} \end{pmatrix}\begin{pmatrix} \hat{q}_i^T & -\hat{q}_i^{\perp T} \end{pmatrix}$ $\mu_s=\sum_i w_i \hat{p}_i \hat{p}_i^T$	$f_s(v)=\sum_i \hat{q}_i\left(\frac{1}{\mu_s}A_i\right)+q_*$ $A_i=w_i \begin{pmatrix} \hat{p}_i \\ -\hat{p}_i^{\perp} \end{pmatrix}\begin{pmatrix} v-p_* \\ -(v-p_*)^{\perp} \end{pmatrix}^{\mathrm{T}}$
刚性变换	$M=\frac{1}{\mu_r}\sum_i w_i \begin{pmatrix} \hat{p}_i \\ -\hat{p}_i^{\perp} \end{pmatrix}\begin{pmatrix} \hat{q}_i^T & -\hat{q}_i^{\perp T} \end{pmatrix}$ $\mu_r=\sqrt{\left(\sum_i w_i \hat{q}_i \hat{p}_i^T\right)^2+\left(\sum_i w_i \hat{q}_i \hat{p}_i^{\perp T}\right)^2}$	$f_r(v)=\lvert v-p_*\rvert\frac{\vec{f}_r(v)}{\lvert\vec{f}_r(v)\rvert}+q_*$ $\vec{f}_r(v)=\sum_i \hat{q}_i A_i$