快速结构化图像修补

路 平[1)] 陈敏刚[1),2)] 马利庄[1)] 桑胜举[3),4)]

[1)](上海交通大学计算机科学与技术系，上海 200240) [2)](上海市科技信息中心，上海 200000)

[3)](华东理工大学机械动力学院，上海 200237) [4)](泰山学院信息科学技术学院，泰安 271021)

摘 要 图像修补的目的是对图像中缺失的区域进行修复，或是将图像中的物体抠去并进行背景填充，以取得融合到难以用肉眼分辨的效果。在图像修补的过程中，较大的结构信息是修补的难点。为此提出了一种快速结构化的图像修补算法，该方法将图像修补分为结构修补与纹理填充两个部分，即在用户指定待修补区域与结构曲线之后，首先定义全局最优化能量函数，并用动态规划与置信度传播的算法将其最小化来完成结构修补；然后对剩余的待修补区域通过按行扫描来进行纹理填充，其中对于边界处的点是使用基于样本的修补算法，而对于待修补区域内部的点，则使用快速的加权 Ashikhmin-WL 算法，扫描完成后输出修补后的图像；最后实现了一个快速结构化图像修补系统，并给出一些实验结果，从实验结果中可以看到，该方法的修补流程与算法是有实际应用价值的。

关键词 图像修补 结构传播 纹理合成 基于样本的算法

中图法分类号：TP391.41 **文献标志码**：A **文章编号**：1006-8961(2010)06-931-05

Fast Structural Image Completion

LU Ping[1)], CHEN Mingang[1),2)], MA Lizhuang[1)], SANG Shengju[3),4)]

[1)](*Department of Computer Science and Technology, Shanghai Jiaotong University, Shanghai* 200240)

[2)](*Shanghai Science & Technology Information Center, Shanghai* 200000)

[3)](*School of Mechanical Engineering, East China University of Science and Technology, Shanghai* 200237)

[4)](*College of Information Science & Technology, Taishan University, Tai'an* 271021)

Abstract Image completion has attracted many researchers these years. The goal of image completion is to repair missing region of images, or to remove objects from images and fill the holes using background information, making it hard to distinguish by eyes. However, to repair huge structure is difficult. We divide the process of image completion into two parts. When the user specified the missing region and structure curves, we first define a global energy function; dynamic programming and belief propagation is used to decide the global minimal cost. This step is also called structure propagation and when it is completed, we scan the region left and implement texture synthesis. For the pixels on boundaries we use exemplar-based algorithm to copy and paste by patch; for the pixels inside the region, we employ a fast weighted Ashikhmin-WL algorithm. At last, the completed image is obtained. We construct a fast structural image completion system and get some results. Experimental results show that our algorithm is useful. Our algorithm will also be extended to video completion in the near future.

Keywords image completion, structure propagation, texture synthesis, exemplar-based algorithm

0 引 言

图像修补(image completion)是计算机图形学与计算机视觉领域内一项具有挑战性的难题。在2000 年，Bertalmio 等人提出的一种基于偏微分方程(PDE)的图像修补方法(image inpainting)[1]是通过对图像受损区域附近等照度线方向进行传播来对输

基金项目：国家自然科学基金项目(60873136)；上海市科委项目(07dz15007)；国家高科技研究发展计划(863)项目(2009AA01Z334)

收稿日期：2009-07-20；**改回日期**：2010-02-20

第一作者简介：路 平(1985—)，男。硕士研究生。研究方向为计算机图像处理。E-mail: brooklu@sjtu.edu.cn

入图像中的缺失或受损区域进行修补，该技术适用于修补图像中的小尺度缺损，可用于旧照片修复与受损绘画修复，但对于较大的缺失区域，则会出现模糊现象，无法取得理想的修复效果。之后的图像修补算法都着眼于缺失区域较大的情况，Criminisi 等人在 2003 年提出了基于样本块的图像修补算法[2]，该算法是通过置信度与优先级的迭代递进全局优化来对图像中较大的缺失区域进行修补，该算法能够在修补过程的同时，自动保持重要结构的传播，其输出图像足以获得以假乱真的效果。但是由于自动化的基于样本的修补算法对于较为复杂的结构会出现不连续现象，因此出现了交互式的算法。Drori 提出的算法[3]要求用户指定“重要点”。Sun 等人 2005 年提出的算法[4]则要求用户先手动完成结构信息的勾勒，然后分别对结构以及纹理进行修补。近年来又有一些新的算法基于全局优化的思路以及置信度传播的方法[5-6]，然而这些算法往往计算量较大，其运行时间差强人意。

本文研究并实现了一种快速图像修补方法，即将图像修补过程分为结构修补与纹理填充两个部分。在结构修补阶段，先通过用户的交互确定结构形状，然后采用全局最优化的思想对结构进行修补；在纹理填充阶段，在讨论了经典的 WL 纹理合成算法的优点与缺点的基础上，提出了一种新的改进算法，使得纹理合成算法被有效地运用到图像修补中。整个修补流程如图 1 所示。

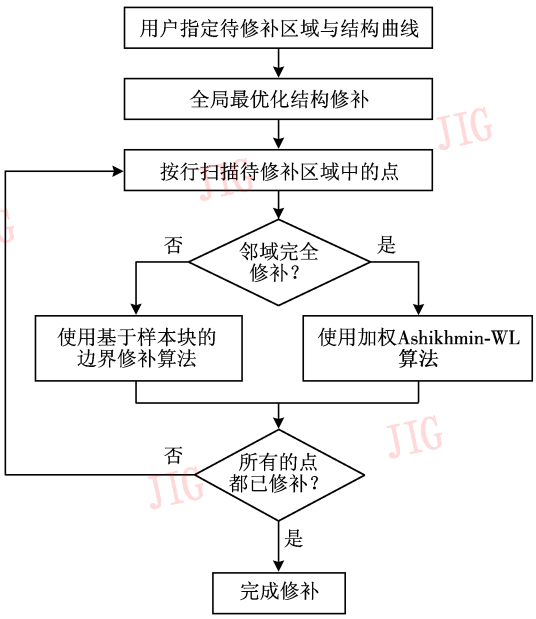


图 1 修补算法流程

Fig. 1 Algorithm for the overall process

1 结构修补

尽管已经有一些自动对缺失区域中的结构信息进行修补的算法，但这些算法仍然无法克服其先天缺陷性，即在修补复杂的结构时仍会出现断层以及不连续情况。因此，本文采用用户交互的方式来得到用户想要的结构信息。事实上，对于用户来说，做这样的交互并未增加多少工作量，有时还能让用户能更好地进行艺术再创作，也能得到更好的修补效果。

如图 2 所示，在确定待修补区域 $\boldsymbol{\Omega}$ 之后，用户再手动指定若干曲线 $\boldsymbol{C}$，在用户完成指定曲线的步骤之后，便可从全部 $\boldsymbol{C}$ 上自动提取一组锚点$\{\boldsymbol{a}_i\}$与一组样本点$\{\boldsymbol{p}_i\}$。这些点彼此间距一般设为样本块宽度的一半，以便获得更好的覆盖区间。其中，锚点来自待修补区域中的曲线，之后的样本块将被贴到这些点上；样本点来自待修补区域外的结构，在结构修补的过程中，样本块将从这些点上提取。样本块的大小可人为设定，在实验中，大多采用 9×9 像素大小。

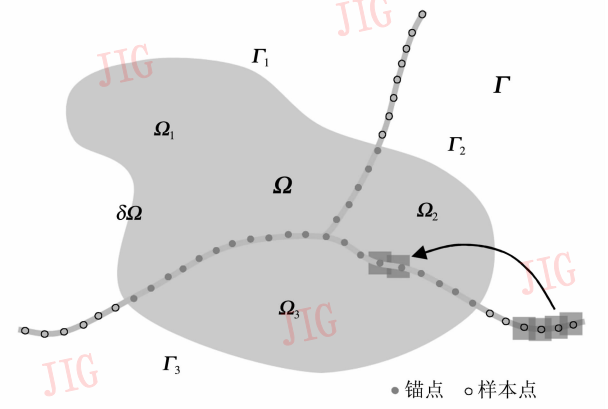


图 2 结构化图像修补示意图

Fig. 2 The sketch map of structural image completion

这样，结构修补的过程就被简化为从样本点选取最合适的样本块贴到锚点上，从而使得所有锚点上的样本块全局最优化的过程。本文定义了一个全局能量函数 $E(X)$，并使用动态规划与置信度传播的方法使其最小化。关于结构修补的详细过程可参考文献[4]。

2 纹理填充

在结构修补完成之后，图像中仍然有大块的待修补区域。由于这些区域中不再含有结构信息，而是被期望为简单的重复性纹理，因此对于这些区域

的修补,就可以用纹理合成的方法来实现。

关于纹理合成有很多研究[7-10]。Wei 和 Levoy 在 2000 年提出了一种简单而经典的 WL 算法[11],该算法按扫描线方向逐个按像素对图像进行修补,其速度成为最大的瓶颈,即使在加入加速算法之后,要合成一幅 200×200 像素大小的图片也需要几分钟到十几分钟的时间。Ashikhmin 提出的改进的 WL 算法[12]极大地提高了效率,而且在修补的过程中保留了较大的纹理块,并避免了 WL 算法会导致的模糊现象,但是该算法在引入到图像修补中时,对于边界无法应用。

本文设计了一种扫描线纹理填充过程,即首先建立修补信息数组 $\boldsymbol{R}$,该数组记录了待修补区域中的所有点的样本来源。在纹理填充开始之前,这个数组被初始化为空。当按行扫描到一个待修补点时,首先在 $\boldsymbol{R}$ 中查找其 L 形邻域的修补情况(如图 3 所示),若该邻域已完全被修补,则直接用加权 Ashikhmin-WL 算法;然后通过邻域中的点的位置偏移来获得候选样本点,否则使用基于样本块的修补方法。

1/4	1/4	1/4	1/4	1/4	1/4	1/4	1/4	1/4
1/4	1/3	1/3	1/3	1/3	1/3	1/3	1/3	1/4
1/4	1/3	1/2	1/2	1/2	1/2	1/2	1/3	1/4
1/4	1/3	1/2	1	1	1	1/2	1/3	1/4
1/4	1/3	1/2	1	p				

图 3　邻域中的权值

Fig. 3　Weights in neighborhood

2.1　加权 Ashikhmin-WL 算法

本文中的邻域 N 指的是以待修补点 $\boldsymbol{p}$ 为中心的样本块 $\boldsymbol{P}$ 中位于点 $\boldsymbol{p}$ 左方或上方的一个 L 形区域内所有点的集合(如图 4 所示,待修补区域 $\boldsymbol{\Omega}$ 内的某一个点的 3×3 大小的 L 形邻域有 4 个点)。当待修补点 $\boldsymbol{p}$ 的邻域已被完全修补时,则该邻域中的每个点都来自于原图中的某个位置,且这个位置被记录在 $\boldsymbol{R}$ 中;然后通过邻域中的点相对于点 $\boldsymbol{p}$ 偏移后的位置即可得到点 $\boldsymbol{p}$ 的一个候选样本点,例如,当位于点 $\boldsymbol{p}$ 左上方的样本点来自于原图中点 $\boldsymbol{q}$ 的位置,则原图中点 $\boldsymbol{q}$ 右下方的点,如果不在待修补区域中,就可以作为点 $\boldsymbol{p}$ 的一个候选样本点。

当邻域大小为 3×3 pixel 时,可产生最多 4 个这样的样本点(可能有的样本点是重复的),在实验

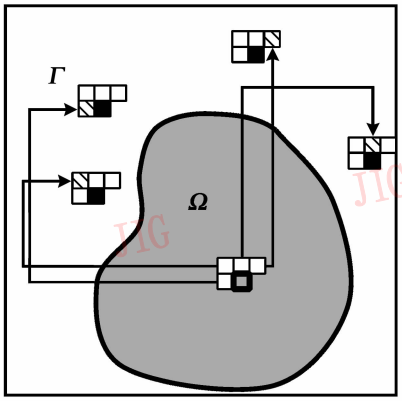


图 4　3×3 L 形邻域

Fig. 4　3×3 L neighborhood

中,通常采用 9×9 pixel 的样本块,这样的邻域大小能产生最多 40 个样本点。这些样本点被放在一个候选样本点集 $\boldsymbol{S}$ 中。

$\boldsymbol{S}$ 被确定之后,再逐个遍历 $\boldsymbol{S}$ 中的样本点,并选出与点 $\boldsymbol{p}$ 的邻域最匹配的点。两个邻域 $\boldsymbol{N}_o$,$\boldsymbol{N}_s$ 的匹配度可以用下述公式来计算:

$$d(\boldsymbol{N}_o,\boldsymbol{N}_s)=\sum_{n\ \mathrm{in}\ N}\big((R_o(n)-R_s(n))^2+(G_o(n)-G_s(n))^2+(B_o(n)-B_s(n))^2\big) \tag{1}$$

其中,$R(\boldsymbol{p})$,$G(\boldsymbol{p})$,$B(\boldsymbol{p})$ 表示该点分别在 RGB 通道上的分量。两个样本块越匹配,其 $d(\boldsymbol{N}_o,\boldsymbol{N}_s)$ 就越小。

由于邻域中的点离点 $\boldsymbol{p}$ 越远,与点 $\boldsymbol{p}$ 的关联度就越弱,因此可为邻域中的点加上权值 ω(如图 3 所示,最内层有最大的权值 1,次内层的权值为 1/2,依此类推)。由此即得到以下新的公式:

$$d(\boldsymbol{N}_o,\boldsymbol{N}_s)=\sum_{n\ \mathrm{in}\ N}\omega_n\big((R_o(n)-R_s(n))^2+(G_o(n)-G_s(n))^2+(B_o(n)-B_s(n))^2\big) \tag{2}$$

接下来,在 $\boldsymbol{S}$ 中找到点 $\hat{\boldsymbol{p}}_s$,使得该点邻域与点 $\boldsymbol{p}_o$ 的邻域相似度最大:

$$\hat{\boldsymbol{p}}_s=\arg\min(d(\boldsymbol{N}_o,\boldsymbol{N}_s)) \tag{3}$$

然后,点 $\boldsymbol{p}_o$ 的颜色值被设为点 $\hat{\boldsymbol{p}}_s$ 的颜色值,同时将修补信息数组 $\boldsymbol{R}$ 中点 $\boldsymbol{p}_o$ 的修补信息设为点 $\hat{\boldsymbol{p}}_s$ 的坐标。这样就完成了该点的修补过程。

2.2　基于样本块的边界修补算法

当待修补点的邻域在修补信息数组 $\boldsymbol{R}$ 中没有被完全记录时,则说明该点位于边界 $\delta\boldsymbol{\Omega}$ 上,无法使用加权 Ashikhmin-WL 算法,而如果用 WL 算法的

话,则会严重影响算法的运算速度。因此需采用一种基于样本块的修补方式。在之前的用户手动交互的过程中,待修补区域已经被划分为若干部分(如图 2 所示)。本文把这些区域标记为 $\boldsymbol{\Omega}_i$。在逐点扫描待修补区域 $\boldsymbol{\Omega}_i$ 的过程中,一旦检测到未修补点,即在与其编号相同的已知区域 $\boldsymbol{\Gamma}_i$ 中寻找最匹配的样本块。这种分治的策略可有效减少寻找匹配样本块时的搜索范围。

两个样本块 $\boldsymbol{P}_1$,$\boldsymbol{P}_2$ 的匹配度可以用类似式(1)的方法来计算,不同的是,在对样本块进行比较时,要对两个样本块中的所有已知区域进行计算,同样的,对于边界 $\delta\boldsymbol{\Omega}$ 上的点 $\boldsymbol{p}$,由于该点上已有样本块 $\boldsymbol{P}_o$,因此可在已知区域 $\boldsymbol{\Gamma}_i$ 中寻找样本块 $\hat{\boldsymbol{P}}_s$,使得

$$\hat{\boldsymbol{P}}_s = \arg\min_{\boldsymbol{P}_s \in \boldsymbol{\Gamma}_i}(d(\boldsymbol{P}_o, \boldsymbol{P}_s)) \tag{4}$$

在 $\hat{\boldsymbol{P}}_s$ 确定之后,不同于加权 Ashikhmin-WL 算法只复制一个点,基于样本块的边界修补算法是将整个 $\hat{\boldsymbol{P}}_s$ 复制到 $\boldsymbol{P}_o$,这样不但提高了效率,也能达到保持纹理中的较大物体的特征的目的;然后再为 $\boldsymbol{R}$ 中所有被复制的点的修补信息赋值。

虽然每个已知区域 $\boldsymbol{\Gamma}_i$ 相比起 $\boldsymbol{\Gamma}$ 来都要小一些,但由于其仍是一个很大的点集,而且每个点都能提取一个样本块,所以边界修补算法比起加权 Ashikhmin-WL 算法来要慢很多,这也是本文不在待修补区域的内部使用边界修补算法的原因之一。

2.3 纹理填充流程

1)建立修补信息数组 $\boldsymbol{R}$ 并初始化为空;

2)按行扫描待修补区域 $\boldsymbol{\Omega}$ 中的点,对于每个未修补的点,判断其邻域情况;

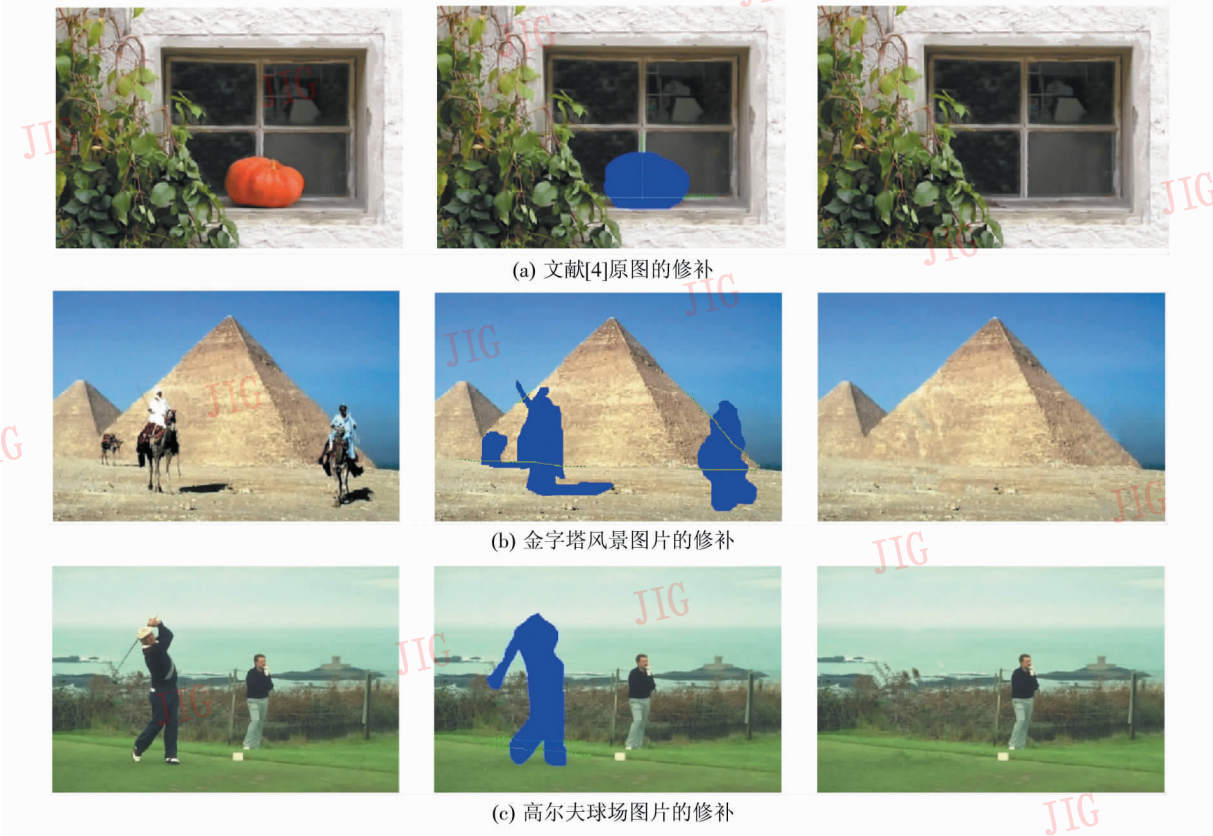
3)如果邻域要完全修补,则使用加权 Ashikhmin-WL 算法进行修补,否则使用基于样本块的边界修补算法进行修补;

4)重复上述步骤直到 $\boldsymbol{\Omega}$ 被完全修补。

3 实验结果

实验环境为 2.4GHz PC,在本文的大部分实验中,样本块被设为 9×9 像素大小,在处理较大物体的图像时,样本块的宽度可以设得更大一些。在给出的例子中,本文的算法在大多数时候运行时间在几十秒到 1 min 之间。

图 5 第 1 行的原图来自文献[4],这个例子很好地说明了交互式图像修补的长处和解决复杂结构



(a) 文献[4]原图的修补

(b) 金字塔风景图片的修补

(c) 高尔夫球场图片的修补

图 5 一些实验结果

Fig. 5 Some results based on our algorithm

的情况。原图中的南瓜所处位置有一十字形结构。修补结果表明,利用交互式的结构修补可以确定窗棂交点的位置,最终给出修补后图像。与文献[4]中的修补结果相比较,为了降低算法的运行时间,本文缩小了搜索空间,而付出的代价就是窗棂左下角处出现的修补瑕疵。本例的修补时间在 10 s 左右。

图 5 第 2 行展示了图像修补的一个应用前景。原图的背景是金字塔。把图中的人物去掉之后,就成了一幅可以用作金字塔景观介绍的风景图片。这样的模式可以用在很多照片上。

图 5 第 3 行是一个比较难的例子。原图中打球的人身后有天空、灌木、草地这 3 块明显不同的区域。而天空和灌木之间的界限比较难以划分,用户无法手动描绘曲线。这揭露出交互式结构修补所受到的限制。即使如此,在输出的图像中还是可以看到在天空和灌木交界处仍能够得到较为合理的修补结果。此结果还有待算法的进一步完善。

4　结　论

本文采用了结构-纹理的图像修补框架思路。在结构修补阶段,类似文献[4]中全局最优化的算法与思想,可将待修补区域划分为若干子区域,以便缩小纹理填充的搜索空间。在纹理填充阶段,不同的子区域,可针对边界和内部的不同情况提出不同的修补算法,以达到快速进行图像修补的目的。接下来的研究目标是改进原算法的不足与提高原算法运行速度,以及研究更健壮的完全自动的结构修补算法。最后类似于文献[13]中的做法,可尝试把本文中的图形修补算法扩展到视频修补领域。

参考文献(References)

[1] Ballester Coloma, Bertalmio Marcelo, Caselles Vincent, et al. Image inpainting[C]//Proceedings of the ACM 27th Annual Conference on Computer Graphics and Interactive Techniques, New York, USA: ACM Press/Addison-Wesley, 2000: 417-424.

[2] Criminisi A, Pérez P, Toyama K. Object removal by exemplar-based inpainting[EB/OL]. http://research. microsoft. com/opps/pubs/default. aspx? id=67273.

[3] Drori I, Cohen-Or D, Yeshurun H. Fragment-based image completion[J]. ACM Transactions on Graphics, 2003, 22(3): 303-312.

[4] Sun Jian, Yuan Lu, Jia Jiaya, et al. Image completion with structure propagation[C]//Proceedings of ACM SIGGRAPH, New York, USA: ACM Press, 2005: 861-868.

[5] Komodakis N, Tziritas G. Image completion using efficient belief propagation via priority scheduling and dynamic pruning[J]. IEEE Transactions on Image Processing, 2007, 16(11): 2649-2661.

[6] Shen J, Jin X, Zhou C, et al. Gradient based image completion by solving the Poisson equation[J]. Computers & Graphics, 2007, 31(1): 119-126.

[7] Jia Jiaya, Tang C K. Image repairing: Robust image synthesis by adaptive nd tensor voting[EB/OL]. http://www. computer. org/portal/web/csdl/doi? doc = abs/proceedings/cvpr/2003/1900/01/190010643abs. htm.

[8] Liang L, Liu C, Xu Y Q, et al. Real-time texture synthesis by patch-based sampling[J]. ACM Transactions on Graphics, 2001, 20(3): 127-150.

[9] Kwatra Vivek, Schödl Arno, Essa Irfan, et al. Graphcut textures: Image and video synthesis using graph cuts[J]. ACM Transactions on Graphics, 2003, 22(3): 277-286.

[10] Hertzmann A, Jacobs C E, Oliver N, et al. Image analogies[C]//Proceedings of ACM SIGGRAPH, New York, USA: ACM Press, 2001: 327-340.

[11] Wei L Y, Levoy Marc. Fast texture synthesis using tree-structured vector quantization[C]//Proceedings of the ACM the 27th Annual Conference on Computer Graphics and Interactive Techniques, New York, USA: ACM Press/Addison-WesleyPublishing Co, 2000: 479-488.

[12] Ashikhmin Michael. Synthesizing natural textures[C]//Proceedings of the 2001 Symposium on Interactive 3D Graphics, New York, USA: ACM Press, 2001: 217-226.

[13] Wexler Y, Shechtman E, Irani M. Space-time completion of video[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(3): 463-476.