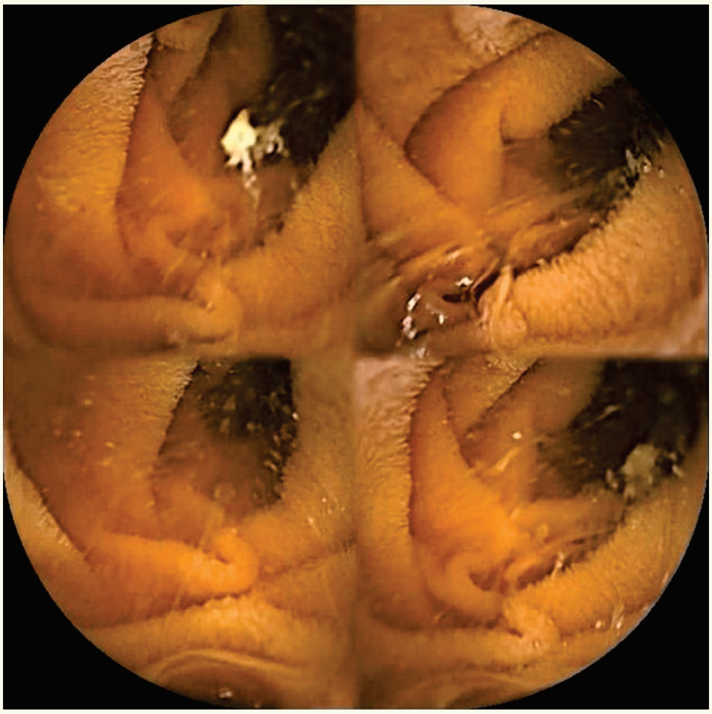
Real-Time Video Texture Synthesis for Multi-Frame Capsule Endoscopy Visualization

Overview

- A common way to shorten the review time of wireless capsule endoscopy is to display multiple video frames simultaneously, side by side.
- Capsule images are nearly circular. Displaying multiple frames requires hole-filling for easier viewing.
- We present a new practical method for real-time video texture synthesis to fill-in such holes.
- Our method is a simplification of Poisson image blending.

Output



Method

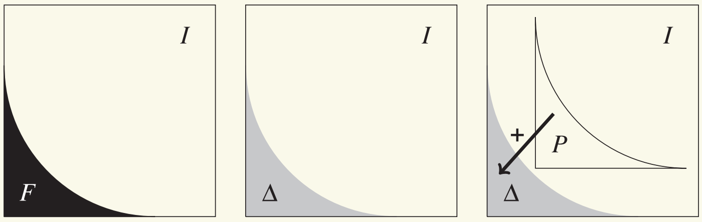


Figure: Texture synthesis of a corner. Left: After warping the image I, we want to fill the empty region F. Middle: First we compute the Δ-map. Right: the fill-in is $F = P + \Delta$, where P is a fixed close image patch. The Δ-map is constructed in-place before adding P.

Algorithm 1: Edge-hiding between image and placed patch

Input : Warped image I, a patch P to place in the fill area F
Output: Δ-map and filled region $F = P + \Delta$ (as in Fig. above)

$\Delta(i,j) \longleftarrow \emptyset$
Compute the distance transform D for each pixel in F from the boundary of I
foreach *pixel* $(i,j) \in F$ *in increasing order of* $D(i,j)$ **do**
  if $D(i,j) < 2$ **then**
    $m(i,j) = mean\big(I(i_1,j_1)$ *such that* $(i_1,j_1) \in I, |i-i_1| \le 1, |j-j_1| \le 1\big)$
    $\Delta(i,j) = m(i,j) - P(i,j)$
  else
    $m(i,j) = mean\big(\Delta(i_1,j_1)$ *such that* $|i-i_1| \le 1, |j-j_1| \le 1, \Delta(i,j) \neq \emptyset\big)$
    $\Delta(i,j) = m(i,j) \cdot d,$ *where d is a decay factor*
  end
end
return $F = P + \Delta$

$I_{0,6}$	$I_{1,6}$	$I_{2,6}$	$I_{3,6}$	$I_{4,6}$	$I_{5,6}$	$I_{6,6}$
$I_{0,5}$	$I_{1,5}$	$I_{2,5}$	$I_{3,5}$	$I_{4,5}$	$I_{5,5}$	$I_{6,5}$
$I_{0,4}$	$I_{1,4}$	$I_{2,4}$	$I_{3,4}$	$I_{4,4}$	$I_{5,4}$	$I_{6,4}$
$\Delta_{0,3}$	$I_{1,3}$	$I_{2,3}$	$I_{3,3}$	$I_{4,3}$	$I_{5,3}$	$I_{6,3}$
$\Delta_{0,2}$	$\Delta_{1,2}$	$I_{2,2}$	$I_{3,2}$	$I_{4,2}$	$I_{5,2}$	$I_{6,2}$
$\Delta_{0,1}$	$\Delta_{1,1}$	$\Delta_{2,1}$	$I_{3,1}$	$I_{4,1}$	$I_{5,1}$	$I_{6,1}$
$\Delta_{0,0}$	$\Delta_{1,0}$	$\Delta_{2,0}$	$\Delta_{3,0}$	$I_{4,0}$	$I_{5,0}$	$I_{6,0}$

Figure: Computation of the Δ-map.
In the first stage, we set Δ for the boundary pixels (grayed) so that the addition of the image patch will not create a strong edge. For example, $\Delta_{2,1} = \frac{1}{3}(I_{2,2} + I_{3,2} + I_{3,1}) - I_{5,4}$.
In the second stage, we propagate the Δ values inwards, e.g. $\Delta_{1,0} = \frac{1}{4}(\Delta_{0,1} + \Delta_{1,1} + \Delta_{2,1} + \Delta_{2,0}) \cdot d$.

References

- Peleg S.: *Elimination of seams from photomosaics* (1981).
- Perez P., Gangnet M., Blake A.: *Poisson image editing* (2003).
- Farbman Z., Hoffer G., Lipman Y., Cohen-Or D.,Lischinski D.: *Coordinates for instant image cloning* (2009).

Ady Ecker
adyecker @ cs.toronto.edu