ParXII: Optimized, Data-Parallel Exemplar-Based Image Inpainting

Mohamed Yousef\(^1\), Khaled F. Hussien\(^2\)
Assiut University, Assiut, Egypt

Figures 1-7, Example input images along with inpainting result

1 Introduction

Completing unknown parts of a damaged input image, or removing objects from photographs and replacing them with visually plausible backgrounds is an important task in photo editing and video processing with a wide range of applications from the reconstruction of missing blocks introduced by packet loss during wireless transmission, reversing of impairments, removal of image objects such as logos, stamped dates, text, and persons, to completing panoramas. The problem with most of existing inpainting methods is the balance between efficiency and accuracy, one of the most accurate methods is exemplar-based image inpainting [Criminisi, et al. 2004], the problem with this method is that it’s very slow and inefficient due to the fact that it needs to scan the whole image before inpainting a certain block of pixels (every scan operation is called a query), this makes the algorithm take tens to hundreds of seconds on modern CPUs.

An important feature of the exemplar-based image inpainting algorithm, along with its accuracy, is that it’s highly parallel; comparisons involved on every query operation can be done completely in parallel, which makes the algorithm map very well on modern massively parallel architectures like modern GPUs. In this paper, we present ParXII, an optimized data-parallel variant of the exemplar image inpainting that seeks to achieve best utilization of today GPUs and then applies a number of optimizations to reduce the number of queries and the arithmetic intensity of each query, with approximately no noticeable effect on plausibility of results.

2 Algorithm

ParXII has the same basic steps of exemplar image inpainting, it modifies some of the steps either to parallelize them or to increase their speed, as follows:

1) Instead of sequentially searching the image for matching patches, the comparison of each block with the target block, using SSD, is done in parallel, after that SSD values are reduced in parallel. An important aspect here is that all modifications on the image must be done on the GPU with no GPU-CPU communication other than the reduced SSD values.

2) Instead of the CIE Lab color space we use the YCbCr color space; the idea here is to use only the chroma components (Cb and Cr) in the SSD comparison. The chroma components preserve the color information and are therefore sufficient for getting to the most appropriate block in the input image.

3) A problem with the formula \( P(p) = C(p)D(p)A(p) \), where \( A(p) \) is the area of to be inpainted part of block, associated with point \( p \), is to choose the block with highest-to-be-inpainted area from blocks in top 5%-10% blocks in priority, to compute this efficiently we approximate it trough modifying the original formula as follows: \( P(p) = C(p)D(p)A(p) \), where \( A(p) \) is the area of to be inpainted part of block, associated with point \( p \).

3 Implementation

We implemented ParXII in OpenCL, we preferred it over CUDA and other platform specific APIs, as it’s an open, platform-independent API. We used Image ImageMagick library to read, write, and modify images. We used the exemplar image inpainting implementation by Qiushuang Zhang as a basis for our modifications and experimentation till we reached ParXII.

4 Results

In the following table (time in milliseconds) we summarize our results, especially compared with the only current attempt to GPU accelerate, exemplar image inpainting [Kwok et al. 2010]. Our results show that we are on the way to real-time video-inpainting.

<table>
<thead>
<tr>
<th>Result</th>
<th>Fast Query on GTX 280</th>
<th>ParXII on GTX 280</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fig 1 (538x403)</td>
<td>11560</td>
<td>1940</td>
</tr>
<tr>
<td>Fig 2 (206x308)</td>
<td>900</td>
<td>170</td>
</tr>
<tr>
<td>Fig 3 (438x297)</td>
<td>7400</td>
<td>1920</td>
</tr>
<tr>
<td>Fig 4 (628x316)</td>
<td>3680</td>
<td>370</td>
</tr>
<tr>
<td>Fig 5 (700x389)</td>
<td>1640</td>
<td>125</td>
</tr>
<tr>
<td>Fig 6 (200x150)</td>
<td>670</td>
<td>80</td>
</tr>
<tr>
<td>Fig 7 (700x438)</td>
<td>4180</td>
<td>430</td>
</tr>
</tbody>
</table>

5 References


---

1 mohamed.mahdi@compit.au.edu.eg
2 khaled.hussain2000@gmail.com