Massively Parallel Layout Generation In Real Time

Vineet Batra
Adobe
vbatra@adobe.com

Ankit Phogat
Adobe
phogat@adobe.com

Tarun Beri
Adobe
tberi@adobe.com

Figure 1: Left image is the input layout; five images on the right show layouts produced by our method

ABSTRACT

Conceiving an artwork requires designers to create assets and organize (or layout) them in a harmonious, self-orating story. While creativity is fundamental to both aspects, the latter can be bolstered with automated techniques. We present a first true SIMD formulation for the layout generation and leverage CUDA-enabled GPU to scan through millions of possible permutations and rank them on aesthetic appeal using weighted parameters such as symmetry, alignment, density, size balance, etc. The entire process happens in real-time using a GPU-accelerated implementation of replica exchange Monte Carlo Markov Chain method. The exploration of design space is rapidly narrowed by performing distant jumps from poorly ranked layouts, and fine tuning the highly ranked ones. Several iterations are carried out until desired rank or system convergence is achieved. In contrast to existing approaches, our technique generates aesthetically better layouts and runs more than two orders of magnitude faster.

SIGGRAPH '19 Posters, July 28 - August 01, 2019, Los Angeles, CA, USA

ACM ISBN 978-1-4503-6314-3/19/07.
https://doi.org/10.1145/3306214.3338596

CCS CONCEPTS

• **Computing methodologies** → *Image-based rendering*;

KEYWORDS

Graphic Design, Page Layout, 2D arrangements

ACM Reference format:
Vineet Batra, Ankit Phogat, and Tarun Beri. 2019. Massively Parallel Layout Generation In Real Time. In *Proceedings of SIGGRAPH '19 Posters, Los Angeles, CA, USA, July 28 - August 01, 2019,* 2 pages.
https://doi.org/10.1145/3306214.3338596

1 INTRODUCTION

Optimally arranging a set of objects in a given space finds applications in several domains. Among others, researchers have examined furniture layout in 3D space [Merrell et al. 2011; Yu et al. 2011] and graphic asset layout in 2D space [O'Donovan et al. 2015]. Prior art, however, points to similar observations and problems. The most prominent one being the inordinately large size of search space (of arrangements), which is computationally prohibitive to fully explore. At the same time, relevant literature finds randomized search space exploration with Monte Carlo Markov Chain [Hastings 1970] methods particularly suitable. Such methods involve optimizing a cost function that measures a layout on a predefined set of weighted parameters (for example, symmetry and balance of a page layout in our case). However, there are two subtle concerns; first, evaluating a layout should be inexpensive and second, on instantiating a poor

layout sample, the system should be able to quickly generate a distant layout efficiently. While the first concern has generally been addressed by accelerating the computations on a programmable graphics device, the second has led to improvements in defining the parameters and tuning their weights.

Despite hardware acceleration, we find the current state-of-the-art solution impractical from a performance standpoint. This is primarily because the proposed algorithm has several sequential steps and fails to utilize GPU optimally. Moreover, the random sampling method produces many bad layouts (e.g overlapping elements), resulting in a large number of false positives being explored.

Our solution improves upon both these aspects by maximizing their data parallelism. To reduce the number of false positives, we make each thread (in a CUDA warp) own a graphic asset and produce a layout sample by modifying a single, randomly chosen parameter (e.g. size, position, etc.) at a time. Thus, multiple layout samples are simultaneously evaluated for validity. At the end, we employ a novel combination of CUDA intrinsics (including 'ballot' and 'random rotate') to quickly propose a valid layout sample. Note, this contrasts existing approaches, which evaluate one permutation at a time and repeat this step till a valid layout is found.

Our technique computes a weighted parameter score for the generated layout sample. All parameters are evaluated in parallel and individual parameter scores are reduced to a weighted sum by employing highly efficient warp intrinsics ('shuffle', 'XOR'). Again, this contrasts existing techniques where individual parameters are mostly evaluated sequentially, leaving the GPU under-utilised.

2 METHOD

Ours is an iterative technique with every iteration trying to better the layout found in the previous one. An iteration works in two stages: 'proposal' and 'evaluation'. The proposal stage aims to find a valid layout that serves as input to the next stage. We generate multiple proposals in parallel. Each proposal chooses a random asset in the artwork and a random operation (i.e. a parameter to be modified and the magnitude of modification). The magnitudes are sampled from a normal Gaussian distribution conditioned on the temperature chain. All proposals are evaluated for validity (in parallel) and a random proposal from the valid ones is chosen.

The second stage ('evaluation') accepts or rejects the chosen proposal depending upon its potential to generate a better layout. For this, its weighted sum of constituent parameters like symmetry, balance, etc. computed in parallel. If the resultant score is better than the current score, the proposal is accepted. Otherwise, an evaluation is done using Metropolis Hastings Criterion [Hastings 1970] conditioned on temperature, which increases the probability of accepting inferior layouts on higher temperatures in the chain, thereby avoiding local minima.

2.1 SIMD Formulation

We create a logical work-group of CUDA threads with each thread dedicated to one asset in layout. The size of a work-group is limited by CUDA warp size (i.e. 32). So, if there are more than 32 assets, we make each thread cater to an equally divided subset 'S' of assets. On the other hand, if there are fewer than 16 assets, we pack multiple work-groups in the same warp.

Each CUDA thread block assigns an operation magnitude ($\sim N(\mu, \sigma^2)$) to each work-group (or warp). The pseudo code executed by every warp (of a thread block) is given below:

```
For each thread in a CUDA block (in parallel):
If first thread in the warp
       Randomly select an operation
       Broadcast the selected operation to other
           ↪ threads in the warp (__shfl intrinsic)
Randomly select an asset from thread's set 'S'
Find magnitude 'M' of operation (normal distribution)
Apply the operation (with magnitude 'M') on the asset
Check validity of the output design and set a bitflag
    ↪  'B' to true if valid, false otherwise
Accumulate 'B' of all threads (__ballot intrinsic)
Mask off bits in other warps
If first thread in the warp
       Randomly select one out of all set bits in 'B'
           ↪  (using random rotate and pick first)
```

3 RESULTS

We compare our results with [O'Donovan et al. 2015] and observe a performance improvement of more than two orders of magnitude. This is predominantly attributed to two things. First, our true SIMD formulation exploits GPU better. Second, we generate 97% valid layout proposals (post randomization) in contrast to 67% valid layouts generated by [O'Donovan et al. 2015]. Finally, as observed by multiple designers, our technique produces qualitatively better layouts than the existing approach. Figure 2 shows a few layouts produced by our method.

Figure 2: Various layouts generated by our method

REFERENCES

Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. Interactive furniture layout using interior design guidelines. *ACM Trans. Graph.*, 30(4): 87:1–87:10, July 2011. ISSN 0730-0301.

Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher. Make it home: Automatic optimization of furniture arrangement. *ACM Trans. Graph.*, 30(4):86:1–86:12, July 2011. ISSN 0730-0301.

Peter O'Donovan, Aseem Agarwala, and Aaron Hertzmann. Designscape: Design with interactive layout suggestions. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1221–1224, 2015. ISBN 978-1-4503-3145-6.

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 04 1970. ISSN 0006-3444. doi: 10.1093/biomet/57.1.97.