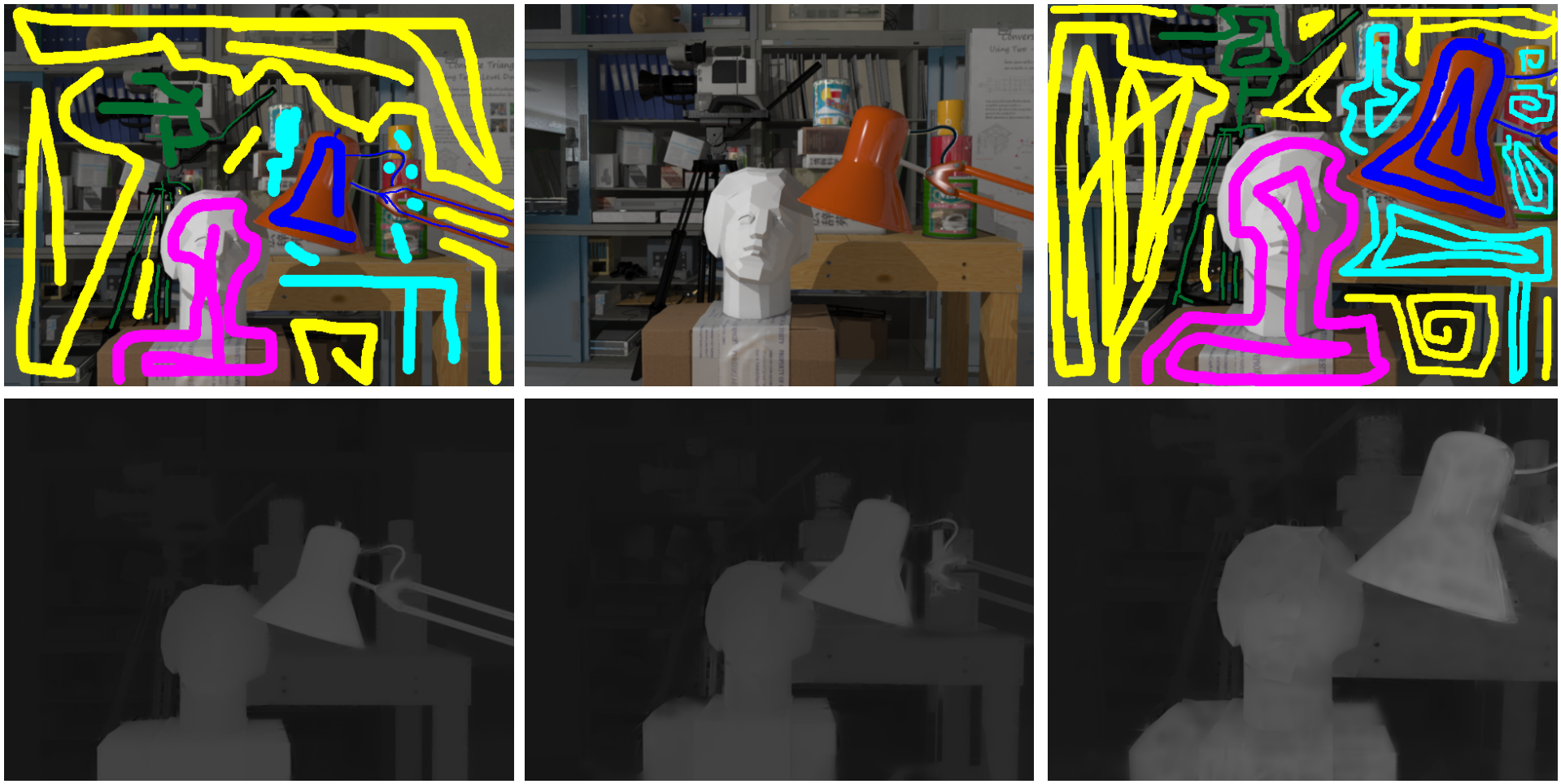
Supplementary Material

Efficient Depth Propagation in Videos with GPU-acceleration

Frame 1 Frame 30 Frame 100



Example result of proposed 2D-to-3D conversion (*Tsukuba1*).

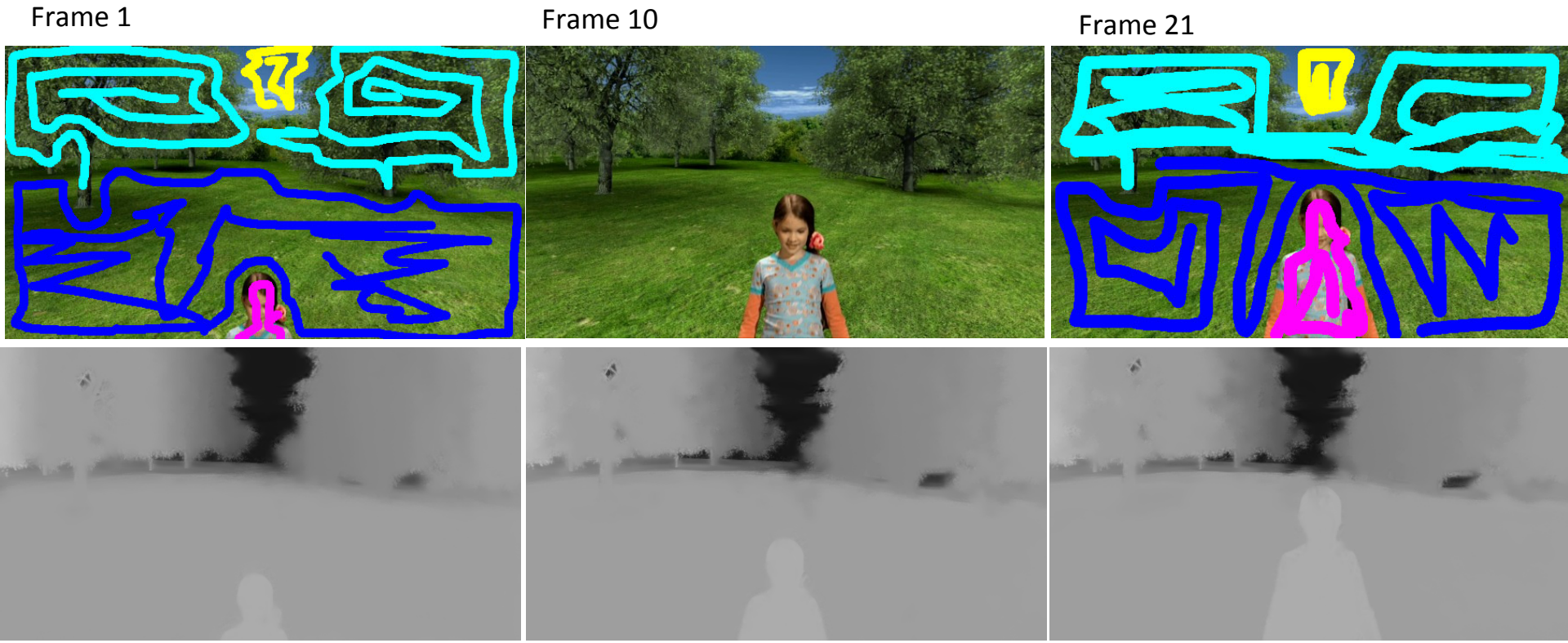
Top: Input video and user input; The depth scribbles in frame 1 and frame 100 are color coded.

Bottom: 2D-to-3D conversion result.

near far near far

Supplementary Material

Efficient Depth Propagation in Videos with GPU-acceleration



Example result of proposed 2D-to-3D conversion (*Child*).

Top: Input video and user input; The depth scribbles in frame 1 and frame 21 are color coded.

Bottom: 2D-to-3D conversion result.

Supplementary Material

Efficient Depth Propagation in Videos with GPU-acceleration

Video / MSE	[1] with OF	[1] without OF	[4]	**Ours, with OF**	**Ours, without OF**
Palace	1.01	1.39	8.51	**0.17**	**0.17**
Parade	0.23	0.23	6.87	**0.12**	**0.13**
City	0.32	0.56	10.30	**0.08**	**0.09**
Football	0.23	0.33	4.95	**0.08**	**0.10**
Stairs	0.19	0.31	1.56	**0.12**	**0.12**

Quantitative evaluation and comparison to [1] and [4].

*MSE*s (mean squared errors) are averaged over all frames and multiplied by 100. Results for [1] and [4] are taken from [1]. For this evaluation the same video, user-input and disparities as in [1] were used to generate resulting disparity maps. For the not optimized algorithm, i.e., [1], and our optimization the results for the algorithm with and without usage of optical flow (*OF*) are listed. All tested algorithms use the same method to estimate the used OF fields, which makes the evaluation independent of the quality of the performed motion estimation.

Supplementary Material

Efficient Depth Propagation in Videos with GPU-acceleration

Video / Runtime	Resolution	Not optimized [1] (C++)	**Our optimization of [1]** (CUDA)
City	699 x 232 x 19	440	**12**
Parade	689 x 282 x 11	286	**7**
Palace	702 x 278 x 10	267	**7**
Stairs	702 x 279 x 10	590	**12**
Football	669 x 282 x 20	974	**27**
Child	600 x 338 x 21	530	**28**
Tsukuba50	640 x 480 x 17	729	**15**
Tsukuba380	640 x 480 x 18	705	**16**

Computational efficiency evaluation of the joint segmentation and propagation step in the un-optimized [1] and our optimized 2D-to-3D conversion algorithm.

The table lists the runtimes in seconds. In this comparison, the joint segmentation and propagation step of the un-optimized and our optimized algorithm take the same input videos, user-input, disparities and parameters.

Supplementary Material

Efficient Depth Propagation in Videos with GPU-acceleration

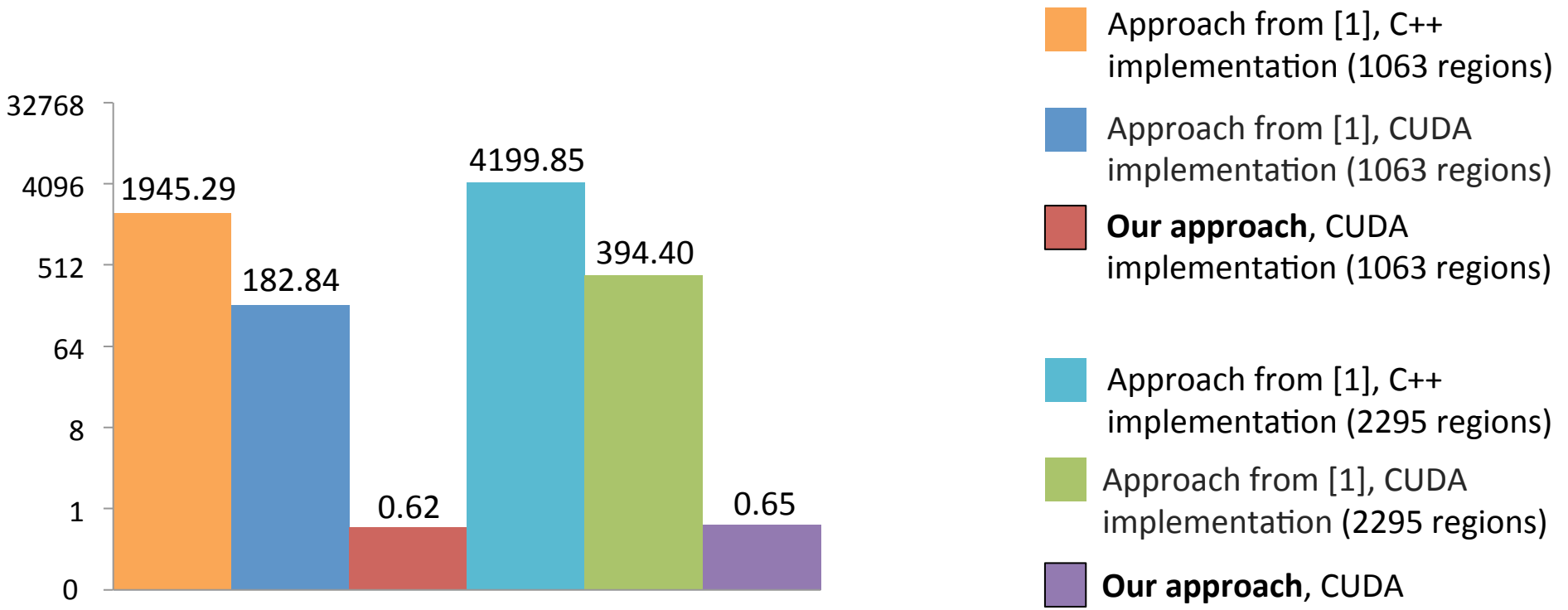
Video / Runtime	Video resolution	**Our optimized approach** (CUDA)	Approach from [1] (CUDA)
City	699 x 232 x 19	**0.62**	1.36
Parade	689 x 282 x 11	**0.14**	0.36
Palace	702 x 278 x 10	**0.13**	0.34
Stairs	702 x 279 x 10	**0.22**	0.75
Football	669 x 282 x 20	**0.44**	1.70
Child	600 x 338 x 21	**1.06**	2.15
Tsukuba50	640 x 480 x 17	**0.27**	1.11
Tsukuba380	640 x 480 x 18	**0.36**	1.12

Computational efficiency evaluation of region-wise filtering (i.e., regularization step) in the un-optimized [1] and our optimized 2D-to-3D conversion algorithm.

The table lists the runtime in seconds. To emphasize the algorithmic contribution in this optimization, both region-wise filtering approaches were implemented in CUDA. In [1] each region is filtered in a separate filtering step. Contrary, our approach filters all regions separately but in a single filtering step. Both approaches applied on the same videos and number of regions.

Supplementary Material

Efficient Depth Propagation in Videos with GPU-acceleration

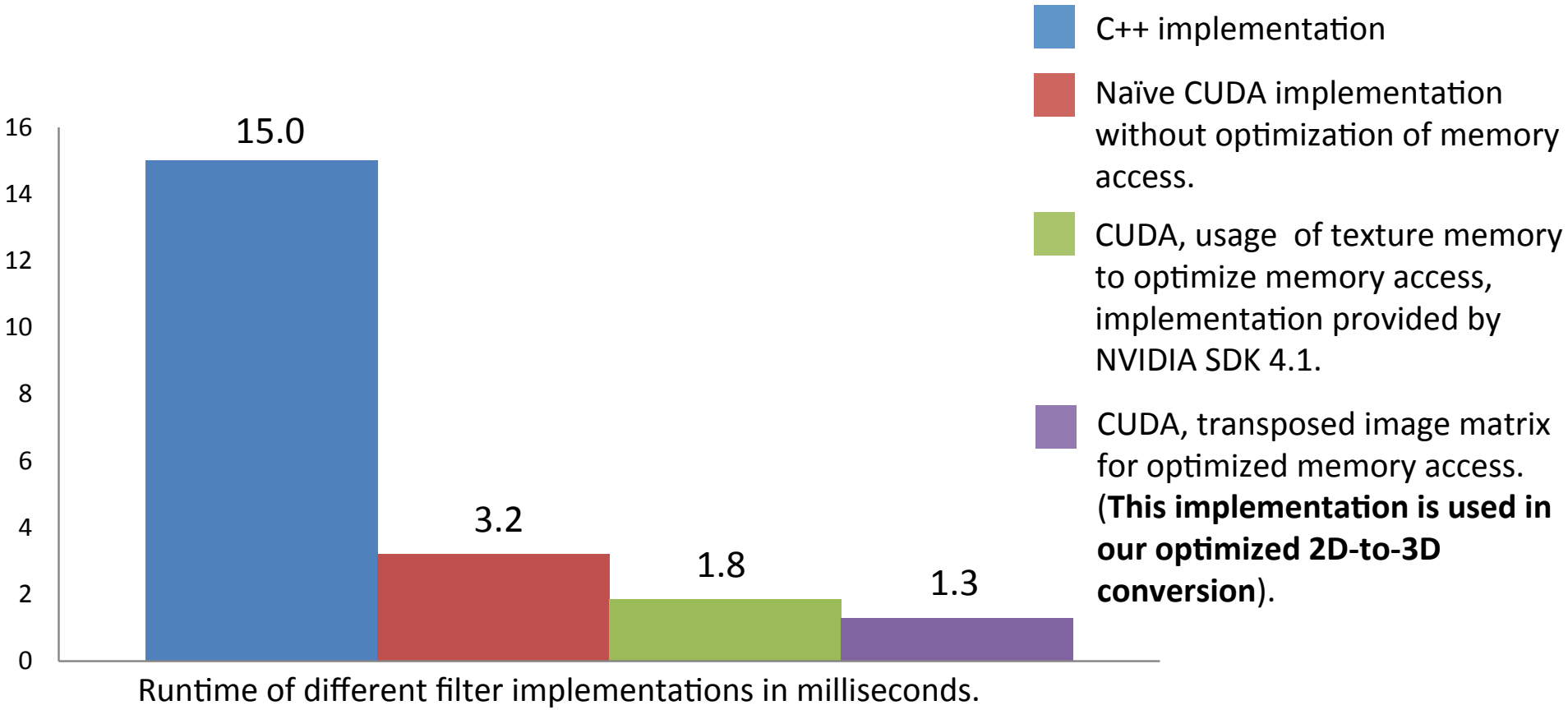


Runtime of different region-wise filter implementations in seconds.

Computational efficiency evaluation of region-wise filtering (i.e., regularization step).
Comparison of runtimes between the C++ and CUDA implementation of the initial region-wise filtering approach from [1] (i.e., each region is filtered in a separate filtering step) and our CUDA implementation of our region-wise filtering (i.e., all regions are separately filtered in one filtering step). The comparison is performed for a segmentation into 1063 and into 2295 regions. The chart lists the runtime of different implementations in seconds for a video with a resolution of 600 x 255 x 20 pixel.

Supplementary Material

Efficient Depth Propagation in Videos with GPU-acceleration



Runtime of different filter implementations in milliseconds.

Computational efficiency evaluation of different box filter implementations in CUDA, which is used in the regularization step (i.e., guided filtering) of the 2D-to-3D conversion algorithm.

The chart lists the runtime of different implementations in milliseconds for an image with a resolution of 1024 x 1024 pixel. Note that the guided filter, which is used in the interpolation step of the 2D-to-3D conversion algorithm, is implemented as a series of box filters. Thus, it is vital to efficiently implementation of a box filter to optimize the 2D-to-3D conversion algorithm from [1].