I associate computer graphics with modern art. Inspired by the words, “I am not an artist, but I can paint with algorithms”, from Dr. Li-Yi Wei’s blog, I started the journey of computer graphics two years ago. Now, I have been working with him for more seven months. With the experiences of research on Markov Chain Monte Carlo (MCMC) rendering, and self-learning projects, my interest in conputer graphics has been cemented, and a solid foundation for related research has been laid. I’d like to continue my research in Monte Carlo rendering. My long-term goal is to develop much more efficient rendering algorithms for challenging scenes and to pursue my career as a prefessor or researcher in this field.

I picked up Monte Carlo rendering by himself, came up new ideas, finished the implementation and managed to write a paper under the supervision of Dr. Li-Yi Wei. The paper was submitted to SIGGRAPH 2019. In the paper, I study MLT-type algorithms building on bidirectional path tracing. Let’s define the failure of obtaining a complete mutation path, which may be caused by the failure of tracing the required two sub-paths or the failure of connecting the two sub-paths, as Proposal Failure. All the current MLT-type algorithms don’t distinguish proposal failure paths from normal proposed paths. Though proposal failure paths, as impossible values of the stationary distribution of Markov chain, don’t affect the final equilibrium, they slow down the convergence rate of the Metropolis-Hastings sampler, which means that the related rendering algorithm produces suboptimal image with the same computation. To address this issue, based on MMLT, we propose a novel algorithm, Proposal Failure MLT (PFMLT), which distinguishes proposal failure paths from normal proposed paths and excludes them from the states of Markov chain. With various scenes of challenging lighting, geometry and material tested, PFMLT always produced better results than MMLT did. Furthermore, our method can be easily used to extend other MLT-type algorithms and make them work much more efficiently. For example, when we extended MMLT and RJMLT in Tungsten, a much simpler and faster renderer than PBRT, with our method, we found that our method always makes extended MMLT better than the original MMLT, extended RJMLT better than the original RJMLT, and even makes extended MMLT better than the original RJMLT.

Prior to that, I wrote three renderers for different projects. One of them is written in C++ from scratch for the project of producing the three ray traced animations from the BART (A Benchmark for Animated Ray Tracing) also under the supervision of Dr. Li-Yi Wei. Through self-learning, I conquered problems, like, how to trace animations? how to parse the complex animation description files? how to eliminate the artifacts caused by high-frequency textures?. Particularly, I implemented Muti-Jittered Sampling, MipMap, EWA for texture filtering and Gaussian filter for image reconstruction to high frequency texture artifacts. Dr. Ulf Assarsson, the original author of BART, spoke highly of my work, and linked my source code and the resulting animations in his project page of the paper.