Jeff Tseng (Yang-Che)

Email : jeffyct@cs.ubc.ca
GitHub: jeff999955
Vancouver, BC V6T

Jeff Tseng is a research student at UBC specializing in computer vision and pursuing his master's degree. His main work revolve around Neural Rendering and SLAM, and he is currently working on diffusion models with Gaussian Splatting.

Education

- **The University of British Columbia**
  Master of Science, Computer Science — *09/2024-*
- **The University of British Columbia**
  Research Assistant — *2023-2024*
- **National Taiwan University** — *GPA 3.92/4.0*
  Bachelor of Science, Computer Science and Information Engineering — *2019 – 2023*

Publication

- **3D Gaussian Splatting as Markov Chain Monte Carlo** — NeurIPS 2024

  Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Jeff Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, Kwang Moo Yi
- **PointNeRF++** — ECCV 2024
  A multi-scale, point-based Neural Radiance Field

  Weiwei Sun, Eduard Trulls, Yang-Che Tseng, Sneha Sambandam, Gopal Sharma, Andrea Tagliasacchi, Kwang Moo Yi
- **Orbeez-SLAM** — ICRA 2023
  A Real-time Monocular Visual SLAM with ORB Features and NeRF-realized Mapping

  Chi-Ming Chung, Yang-Che Tseng, Ya-Ching Hsu, Xiang-Qian Shi, Yun- Hung Hua, Jia-Fong Yeh, Wen-Chin Chen, Yi-Ting Chen, Winston H. Hsu

Research Experience

- **Computer Vision Lab, UBC** — Vancouver, BC
  Research Assistant — *2023*
  - Explore the difficulty of 3D Gaussian Splatting on dynamic scenes on datasets such as DynamicScene, HyperNeRF, and DyCheck iPhone.
  - Separately model static and dynamic Gaussians with optical flow masks and discover limitations when camera motion is large.
  - Introduce Video Diffusion Model as a refiner to alleviate the sparse input issue from dynamic datasets.
- **PointNeRF++** — Vancouver, BC
  Research Intern — *2023*
  - Conducted experiments on KITTI-360, ScanNet, and NeRF Synthetic dataset for 3D Gaussian Splatting.
  - Explore the difficulty of 3D Gaussian Splatting on real-world scenes.
- **Orbeez-SLAM (Communications & MultiMedia Lab)** — Taiwan
  Undergraduate Research Student — *2022*
  - Used **CUDA & C++** to migrate instant-ngp NeRF to ORB-SLAM to achieve **3D scene reconstruction** with **800x speedup** over other NeRF-based SLAMs.
  - Derived rendering loss of NeRF in terms of camera pose and verify if the `instant-ngp` source code matches our derivation.
- **Machine & Intelligence Understanding Lab** — Taiwan
  Undergraduate Research Student — *2021 Fall - 2022 Spring*
  - Researched on enhancing Automatic Speech Recognition (ASR) and Spoken Language Understanding through natural language pretraining.
  - Arranged **over 20 cross-modal (text and speech) tasks**, their benchmarks, and datasets.

Employment History

- **GoGoX** Vancouver
  Software Engineering, Data Scientist *Nov, 2022 - Jun, 2024*
  - **Discrete-choice Stopping Model**:
    - Extracted the **Graphhopper** routing engine from the deprecated **JAVA** dispatch system, built the minimal version with **Docker Compose**, and provided a function in Python.
    - Conducted **counterfactual analysis** on linear model which estimates the possibility of a driver to stop working by cumulative income, cumulative working hours, and other driver- or time-dependent factors.
  - **GoGoX QA Bot**:
    - Design the **RAG+LLM** pipeline based on **OpenAI**, **Pinecone**, and **Langchain** to support **customized ChatGPT** with **prompt engineering**.
    - Developed a minimum viable product (MVP) using **Streamlit**, allowing beta users to interact with the chatbot.
    - Designed APIs to capture user feedback and responses, utilizing this data to enable memory-based conversations in the chatbot.
    - Reduced a **2** second latency of Pinecone with a self-hosted **Redis Stack** vector database.
  - **Working Time Labeling**:
    - Developed a linear model and automated the labeling process of pickup and arrival time.
    - Conducted **feature engineering** on driver and order data for the linear model to identify the anomaly behavior of driver to help filter out cases unsuitable for labeling model.
  - **GPS Map Matching**:
    - Employed **Python shapely** to determine which roads the driver is probably driving on the map given a GPS coordinate.
    - Find the route with Hidden Markov Model with **Python** given possible road segments.
    - Built a **visualization tool** with **Python folium** and **OpenStreetMap** to analyze the GPS track and users behavior.
- **ASUS Intelligence Cloud Service** Taiwan
  Software Engineering Intern *Jul 2022 - Sep 2022*
  - **Selenium IDE Modification**:
    - Added popup menu to a Selenium IDE by **Electron** and **Node.js** to support additional recording functions and **facilitate the testing** phase which **enabled 19** out of **23** testcases to be automatically recorded.

Side Projects

- **Clapping Detection**: Examined and cleaned **4** different datasets labeled by actions, developed a model using **PyTorch** with **92%** accuracy on Kinetics dataset, and adapted to a custom dataset of videos featuring children clapping.
- **Draw-Straight-Line Detector** : Developed an algorithm using traditional computer vision techniques such as **contour detection** and **filtering** to determine whether the subject in a video is drawing a straight line, implemented with **OpenCV in C++**.
- **Open AIFab**: Integrated **7** computer vision models implemented by **PyTorch** and **Tensorflow** into **Flask** as API for the backend team.
- **Coughvid**: Integrated audio models by **Pytorch** for cough detection and linear models for clinical data to a **Flask** API designed for COVID-19 cough detection.
- **Japanese Memorizer App**: Developed a vocabulary learning mobile application with **Flutter**, bloc, **firebase backend**, and the **CI/CD** pipeline on codemagic.
- **nvtop modification**: Added user filtering feature, in C, to an open-source GPU monitoring tool - nvtop.
- **LeaST: League Small Tool**: Developed automation tools such as spam clicking and auto-accepting match invitation for League of Legends, implemented using **C++** with **Win32 API** and **Python** with **League Client API**.