BlabberSeg: Semantic Perception for Reliable Open-Vocabulary UAV Safe Landing

Haechan Mark Bong[1,2], Giovanni Beltrame[1,2]

***Abstract*— Reliable robot autonomy requires semantic perception that remains both informative and fast enough for closed-loop safety decisions. We present BlabberSeg, an optimized CLIPSeg-based open-vocabulary segmentation pipeline for UAV emergency landing. The method targets semantic reliability under edge constraints by reusing prompt, positional, and image features and deploying floating-point 16 ONNX (TensorRT) inference. In a DOVESEI-based safe-landing workflow, BlabberSeg reaches 16.78Hz on Jetson Orin AGX (64GB), a 927.41% speed increase over the original CLIPSeg (1.81Hz), with limited degradation in segmentation agreement (2.1% relative area difference) and mIoU (9%). At the task level, safe-landing success is preserved (76/100, matching baseline) while mission time is substantially reduced. These results support semantic open-vocabulary perception as a practical component for reliable autonomous landing.**

I. INTRODUCTION

Semantic perception is central to reliable robot autonomy, especially when control actions depend on scene understanding under uncertainty. In UAV emergency landing, the autonomy stack must repeatedly infer safe landing zones (SLZs) from high-altitude imagery and update decisions fast enough to remain safe during descent.

Open-vocabulary vision-language models (VLMs) provide useful semantic flexibility over fixed-label segmentation, but their compute cost can break real-time safety loops on edge hardware. CLIPSeg [1], built on CLIP [2], is promising for semantic SLZ perception, yet naive deployment is too slow for robust onboard operation, particularly when offloading is unreliable [3].

We address this reliability bottleneck through efficiency-aware semantic perception. This paper emphasizes three contributions:

- **Semantics-aware edge optimization:** prompt and feature reuse to remove redundant CLIPSeg computation while preserving semantic outputs.
- **Reliable onboard runtime:** FP16 ONNX/TensorRT deployment for sustained high-rate inference on Jetson-class systems.
- **Safety-oriented evaluation:** quantitative analysis of semantic consistency, robustness, and task-level safe-landing outcomes in DOVESEI.

Qualitatively, segmentation outputs remain visually close to the baseline, with only small boundary-level deviations (Fig. 1).

[1]Department of Computer Engineering and Software Engineering, Polytechnique Montréal, QC., Canada. haechan.bong@etud.polymtl.ca
[2]MILA, QC., Canada.

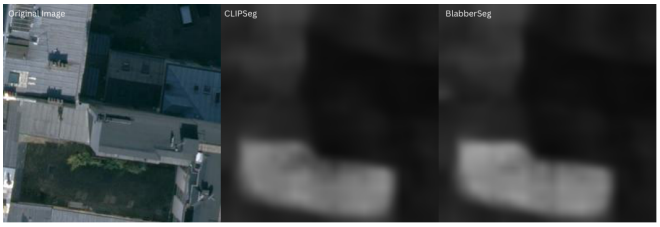

Fig. 1: Visual comparison between baseline CLIPSeg and BlabberSeg masks. Differences are minor and concentrated near class boundaries, consistent with the small quantitative drop reported in Fig. 3.

II. RELATED WORK

A. Edge Acceleration for Foundation Models

Prior acceleration methods include compression [4], [5], filtering/reusing [6], [7], [8], [9], and dynamic early-exit strategies [10], [11]. These methods improve efficiency, but direct transfer to multimodal foundation models can reduce robustness or add heavyweight control heads.

Edge-cloud partitioning has also been explored [3]; however, safety-critical UAV operations in GPS- or network-challenged environments cannot assume stable links. This motivates fully onboard optimization.

B. Safe-Landing Perception in UAVs

Many safe-landing systems operate at low altitude [12], [13], [14], [15], [16]. At 100m altitude, segmentation becomes harder due to low spatial resolution and motion effects. DOVESEI [17] demonstrated open-vocabulary SLZ detection with CLIPSeg, but practical deployment still requires real-time inference.

III. METHOD

A. Overview

BlabberSeg is an optimized CLIPSeg inference stack for repeated aerial segmentation queries on edge AI hardware. Given incoming images and a prompt set describing candidate SLZ classes (e.g., grass, field, park), the system outputs segmentation masks used by a downstream safe-landing controller.

B. Optimization Strategy

(1) Reusing positional embeddings. For fixed input sizes, positional embedding interpolation is precomputed once and reused across frames, removing per-frame interpolation overhead.

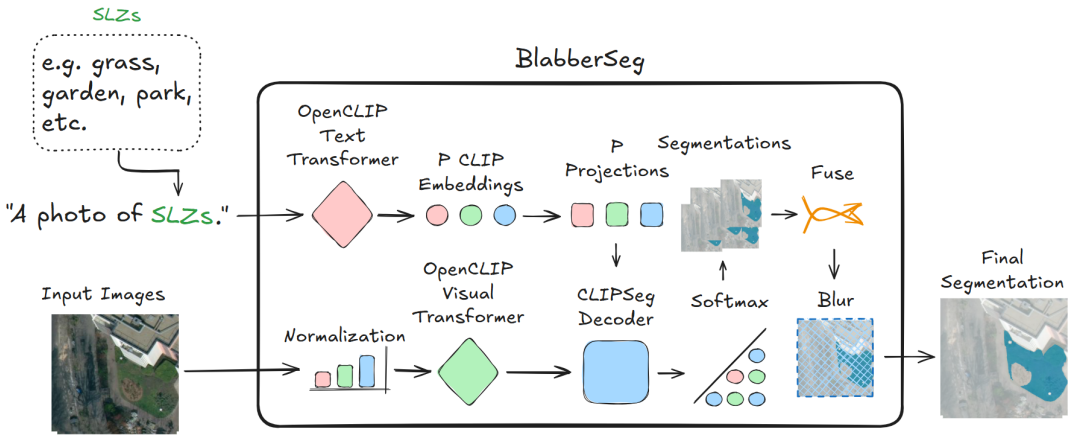

Fig. 2: BlabberSeg architecture for safe aerial robotics. Prompt embeddings, positional embeddings, and selected intermediate features are reused to avoid redundant computation.

(2) Reusing prompt embeddings. Prompt embeddings are cached and recomputed only when the prompt set changes. This is effective for safe-landing missions where semantic targets are stable over short horizons.

(3) Reusing image features. When multiple prompts are applied to the same frame, shared image encoder features are computed once and reused by the decoder.

(4) Embedded runtime optimization. The model is converted to floating-point (FP) 16 and deployed through ONNX (TensorRT) with input-output binding to reduce latency and memory transfer overhead.

Together, these changes preserve CLIPSeg behavior while reducing redundant work in the exact regime required by safe aerial robotics: repeated inference on related images/prompts under tight onboard constraints.

IV. EXPERIMENTAL SETUP

A. Hardware and Data

We evaluate on Jetson-class platforms and report primary results on NVIDIA Jetson Orin AGX 64GB. For segmentation consistency analysis, we use 500 aerial images sourced from Google Maps [18] and compare optimized outputs against the original CLIPSeg output as reference.

B. Safe-Landing Evaluation

To connect model optimization to robotic outcomes, we integrate BlabberSeg into DOVESEI [17]. The simulator performs emergency descent from 100m to 20m and repeatedly selects SLZs based on segmentation and visual-servoing logic.

C. Metrics

We report: (i) segmentation duration / frequency (Hz), (ii) segmentation consistency (accuracy ratio and mIoU vs. CLIPSeg reference), and (iii) safe-landing task metrics (successful SLZ selections, mission time, and horizontal travel distance).

V. RESULTS AND DISCUSSION

A. Real-Time Segmentation on Embedded Hardware

On Jetson Orin AGX 64GB, BlabberSeg improves throughput from 1.81Hz to 16.78Hz (927.41% increase), enabling real-time use in landing loops. The optimized system keeps high agreement with the baseline, with only minor reductions in segmentation consistency.

Model Legend:

- **Original**: CLIPSeg (original implementation)
- **FP**: CLIPSeg with FP16
- **RPE**: CLIPSeg + reusing prompt embeddings
- **FP-RPE**: FP16 CLIPSeg + reusing prompt embeddings
- **HF**: CLIPSeg (Hugging Face implementation)
- **HF-RPE**: Hugging Face CLIPSeg + reusing prompt embeddings
- **FP-RPPE**: FP16 CLIPSeg + reusing prompt and positional embeddings
- **FP-RPPET**: FP-RPPE + TensorRT
- **FP-RPPETI**: FP-RPPET + input/output binding

TABLE I: Quantitative results of comparison among original CLIPSeg and optimized CLIPSeg models for NVIDIA Jetson Orin AGX 64G. GT* denotes ground truth (original CLIPSeg).

Model	Mean Segmentation Duration (s)	Mean Accuracy (%)	mIoU
Original	0.553±0.004	GT*	GT*
FP	0.201±0.019	99.5±0.29	99.6±0.36
HF	0.677±0.032	GT*	GT*
HF-RPE	0.617±0.012	GT*	GT*
RPE	0.547±0.044	GT*	GT*
FP-RPE	0.159±0.010	99.5±0.29	99.6±0.36
FP-RPPE	0.108±0.009	99.5±0.01	97.7±1.49
FP-RPPET	0.077±0.006	97.9±1.27	91.0±8.40
FP-RPPETI	0.060±0.009	97.9±1.27	91.0±8.40

On Jetson Orin AGX 64G, optimization yields a clear latency-accuracy trade-off (Table I). The baseline runs at 0.553s progressively reduce duration to 0.060s (FP-RPPETI),

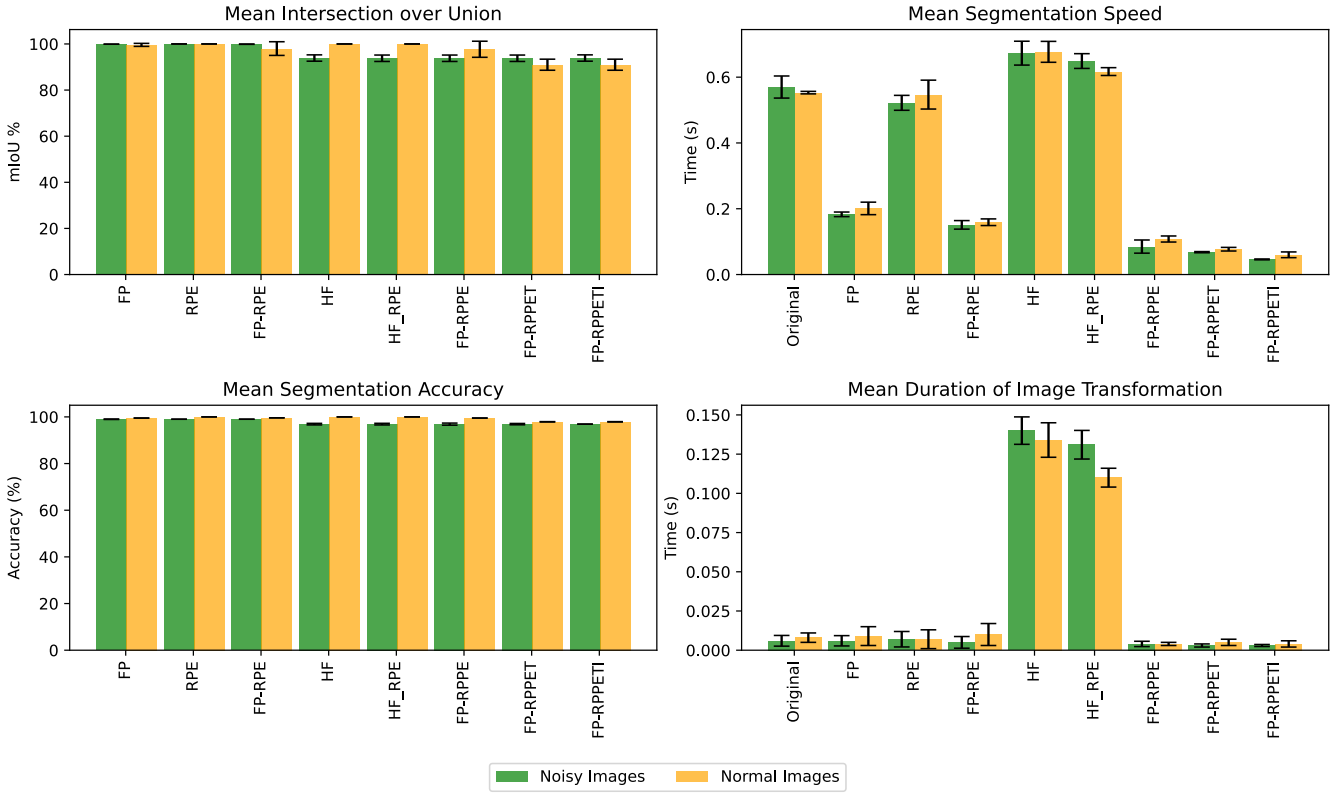

Fig. 3: Noisy-image evaluation: mIoU, transformation time, segmentation accuracy ratio, and speed remain near clean-image trends.

corresponding to the reported real-time gain. Across this progression, segmentation agreement remains high, with mean accuracy around 97.9–99.5% and mIoU around 91.0–99.6, indicating that substantial speedup is achieved with controlled semantic-quality degradation.

B. Impact on Safe-Landing Performance

Despite substantial acceleration, safe-landing quality is preserved in DOVESEI evaluation. Both systems achieve identical successful SLZ selections (76/100), while BlabberSeg reduces average mission time by approximately $8\times$ due to higher perception update rate (Table II).

TABLE II: Safe-landing comparison in DOVESEI simulation.

Metric	DOVESEI	BlabberSeg
Successful SLZ selections	76/100	76/100
Avg. horizontal distance (m)	80.97±20.11	89.12±15.90
Avg. mission time (s)	506.47±49.30	62.72±20.06
mIoU	38±2.33	37±3.49
Avg. inference speed (Hz)	1.90±0.21	16.70±0.19

C. Robustness Under Visual Noise

Figure 3 evaluates robustness using a 100-image random-noise stress test. BlabberSeg remains stable under perturbation: mIoU, segmentation accuracy, and throughput stay close to the clean-image case. These results indicate that the acceleration gains are not obtained at the cost of fragile semantic behavior, which is important for safety-critical autonomous landing.

D. Cross-Device Performance and Scalability

Table III demonstrates that the proposed optimization pipeline scales across heterogeneous edge hardware. On emulated Jetson Orin NX 16G and Orin Nano 8G, the best variant (FP-RPPETI) reaches 14.79Hz and 12.63Hz, i.e., 2205.0% and 2695.2% speedup over each baseline. On Jetson AGX Xavier 64G, BlabberSeg reaches 15.18Hz with a 2383.2% speedup, showing that the same reuse and runtime strategy transfers across NVIDIA generations.

For CPU-only execution on Raspberry Pi 5, optimization still provides substantial relative acceleration (0.02Hz to 0.17Hz, +715.5%). While this remains below strict real-time requirements, it confirms that reuse-based efficiency gains are robust even under highly constrained compute settings.

E. Implications for Reliable Semantic Autonomy

The main outcome is not only higher speed but improved reliability of semantics-driven autonomy. Higher update rates reduce perception staleness during descent, improving responsiveness to scene change. The observed trade-off is practical for safety-critical use: a small semantic-quality drop for a large reduction in decision latency.

Figure 4 highlights GPU power behavior, and Figure 5 highlights thermal behavior during accelerated inference. To-

TABLE III: Quantitative results of comparison among original CLIPSeg and optimized CLIPSeg models for other NVIDIA Jetson devices and Raspberry Pi 5. GT*: ground truth.

Device	Models	Mean Segmentation Duration (s)	Increase in Speed (%)	Hz
NVIDIA Jetson Orin NX 16G (Emulated)	Original	1.490±0.067	GT*	0.67±0.0301
	FP	0.420±0.023	354.7±4.31	2.38±0.1296
	HF	1.537±0.040	97.0±0.62	0.65±0.0185
	HF-RPE	1.469±0.0474	101.5±4.27	0.68±0.1285
	RPE	1.364±0.035	109.2±0.55	0.73±0.0166
	FP-RPE	0.378±0.018	394.5±0.73	2.65±0.0220
	FP-RPPE	0.118±0.008	1258.2±18.95	8.44±0.5705
	FP-RPPET	0.094±0.009	1592.6±35.17	10.69±1.0588
	FP-RPPETI	**0.068**±0.007	**2205.0**±53.72	**14.79**±1.6174
NVIDIA Jetson Orin Nano 8G (Emulated)	Original	2.135±0.033	GT*	0.47±0.0072
	FP	0.594±0.023	359.2±8.91	1.68±0.0643
	HF	2.184±0.017	97.7±2.35	0.46±0.0170
	HF-RPE	2.105±0.016	101.4±8.00	0.48±0.0577
	RPE	2.044±0.071	104.4±0.50	0.49±0.0036
	FP-RPE	0.553±0.018	386.2±0.50	1.81±0.0036
	FP-RPPE	0.150±0.009	1424.7±57.81	6.67±0.4171
	FP-RPPET	0.099±0.007	2147.0±97.29	10.06±0.7019
	FP-RPPETI	**0.079**±0.008	**2695.2**±166.29	**12.63**±1.1997
NVIDIA Jetson AGX Xavier 64G	Original	1.570±0.078	GT*	0.64±0.0317
	FP	0.419±0.048	374.6±8.72	2.39±0.2767
	HF	1.615±0.051	97.2±0.16	0.62±0.0050
	HF-RPE	1.550±0.040	101.3±3.79	0.65±0.1204
	RPE	1.417±0.010	110.8±0.61	0.71±0.0194
	FP-RPE	0.356±0.015	441.1±0.52	2.81±0.0166
	FP-RPPE	0.112±0.021	1404.4±55.58	8.94±1.7636
	FP-RPPET	0.076±0.003	2055.5±13.63	13.09±0.4326
	FP-RPPETI	**0.066**±0.008	**2383.2**±59.78	**15.18**±1.8970
Raspberry Pi 5	Original	42.973±1.536	GT*	0.02±0.0008
	HF	27.641±0.888	155.5±0.49	0.04±0.0004
	HF-RPE	21.154±0.255	203.2±1.40	0.05±0.0012
	RPE	21.770±0.193	197.4±0.68	0.05±0.00006
	RPPE	**6.096**±0.966	**715.5**±33.03	**0.17**±0.0275

gether, they show an additional reliability benefit of efficient computation: lower and more stable GPU power draw and temperature reduce the risk of thermal throttling and energy spikes during mission-critical descent. For safe autonomy, this improves timing predictability of perception updates, supports longer onboard operation under battery constraints, and lowers the chance that compute stress degrades closed-loop landing decisions when the vehicle is already in a safety-critical state.

VI. LIMITATIONS AND FUTURE WORK

Current validation is primarily simulation-based due to legal and safety limits. Outdoor trials in dense urban scenes, diverse weather, and changing illumination remain necessary before deployment claims. Performance on very low-power CPU-only platforms (e.g., Raspberry Pi class) remains insufficient for real-time safe-landing, indicating that lightweight edge accelerators are still required.

Future work will focus on uncertainty-aware SLZ scoring, temporal consistency across frames, and hardware-in-the-loop outdoor experiments with safety monitors.

VII. CONCLUSION

We presented BlabberSeg as a semantic perception component for reliable robot autonomy in UAV emergency landing. By combining feature/prompt reuse with embedded runtime optimization, BlabberSeg increases CLIPSeg from 1.81Hz

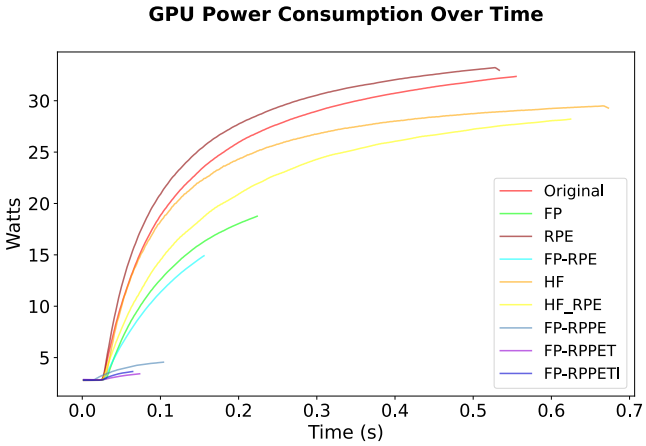

Fig. 4: GPU power consumption during accelerated inference.

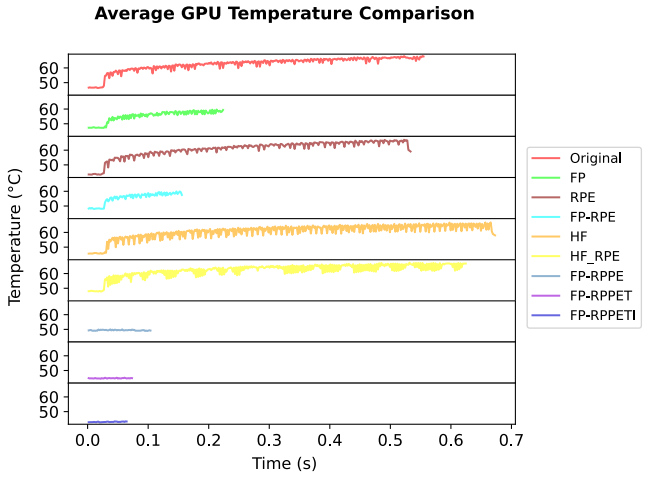

Fig. 5: GPU temperature during accelerated inference.

to 16.78Hz on Jetson Orin AGX while preserving practical safe-landing outcomes in DOVESEI. Overall, the results support open-vocabulary semantics as a feasible and reliability-relevant layer for onboard autonomous safety pipelines.

REFERENCES

[1] T. Luddecke and A. Ecker, "Image Segmentation Using Text and Image Prompts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 7086–7096.

[2] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal *et al.*, "Learning Transferable Visual Models from Natural Language Supervision," 2021.

[3] B. Yang, L. He, N. Ling, Z. Yan, G. Xing, X. Shuai *et al.*, "EdgeFM: Leveraging Foundation Model for Open-Set Learning on the Edge," 2023.

[4] W. Niu, X. Ma, S. Lin, S. Wang, X. Qian, X. Lin *et al.*, "PatDNN: Achieving Real-Time DNN Execution on Mobile Devices with Pattern-based Weight Pruning," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '20. ACM, Mar. 2020.

[5] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized Convolutional Neural Networks for Mobile Devices," 2016.

[6] Z. Hu, N. Ye, and I. Mohomed, "mmSampler: Efficient Frame Sampler for Multimodal Video Retrieval," in *Proceedings of Machine Learning*

and Systems, D. Marculescu, Y. Chi, and C. Wu, Eds., vol. 4, 2022, pp. 153–171.

[7] M. Yuan, L. Zhang, F. He, X. Tong, M.-H. Song, Z. Xu *et al.*, "InFi: End-to-End Learning to Filter Input for Resource-Efficiency in Mobile-Centric Inference," 2023.

[8] U. Drolia, K. Guo, J. Tan, R. Gandhi, and P. Narasimhan, "Towards Edge-Caching for Image Recognition," in *2017 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2017, Kona, Big Island, HI, USA, March 13-17, 2017*. IEEE, 2017, pp. 593–598.

[9] M. Xu, M. Zhu, Y. Liu, F. X. Lin, and X. Liu, "DeepCache: Principled Cache for Mobile Deep Vision," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '18. ACM, oct 2018.

[10] Q. Cao, P. Khanna, N. D. Lane, and A. Balasubramanian, "MobiVQA: Efficient On-Device Visual Question Answering," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 6, no. 2, jul 2022.

[11] I. Leontiadis, S. Laskaridis, S. I. Venieris, and N. D. Lane, "It's always personal: Using Early Exits for Efficient On-Device CNN Personalisation," in *Proceedings of the 22nd International Workshop on Mobile Computing Systems and Applications*, ser. HotMobile '21. ACM, Feb. 2021.

[12] M. K. Mittal, A. Valada, and W. Burgard, "Vision-based Autonomous Landing in Catastrophe-Struck Environments," *ArXiv*, vol. abs/1809.05700, 2018.

[13] E. Chatzikalymnios and K. Moustakas, "Landing Site Detection for Autonomous Rotor Wing UAVs Using visual and Structural Information," *Journal of Intelligent & Robotic Systems*, vol. 104, feb 2022.

[14] C. Forster, M. Faessler, F. Fontana, M. Werlberger, and D. Scaramuzza, "Continuous On-board Monocular-Vision-based Elevation Mapping Applied to Autonomous Landing of Micro Aerial Vehicles," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 111–118.

[15] M. Rabah, A. Rohan, M. Talha, K.-H. Nam, and S. Kim, "Autonomous Vision-based Target Detection and Safe Landing for UAV," *International Journal of Control, Automation and Systems*, oct 2018.

[16] J. Park, Y. Kim, and S. Kim, "Landing Site Searching and Selection Algorithm Development Using Vision System and its Application to Quadrotor," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 488–503, 2015.

[17] H. M. Bong, R. Zhang, R. de Azambuja, and G. Beltrame, "Dynamic Open Vocabulary Enhanced Safe-landing with Intelligence (DOVE-SEI)," 2023.

[18] Google, "Google Maps," 2023, accessed: 2023-08-19.