

Modernizing robotics development by integrating Visual Studio Code and using SSH to deploy code to the KIPR wombat

Dhruv Sunder

Abstract -The advancement of educational robotics requires a development environment that balances hardware accessibility with professional software engineering standards. While the KIPR Wombat controller provides a dedicated integrated development environment (IDE) for ease of use, it lacks the sophisticated features found in industry-standard tools. This paper proposes a hybrid development method that utilizes Visual Studio Code (VS Code) as the primary local editor, synchronized with the Wombat controller through the Secure Shell (SSH) protocol. This approach allows students to leverage advanced version control, real-time static analysis, and AI-assisted coding practices. By maintaining the Wombat's native compilation mechanism, full hardware compatibility is ensured while significantly enhancing developer productivity and workflow efficiency.

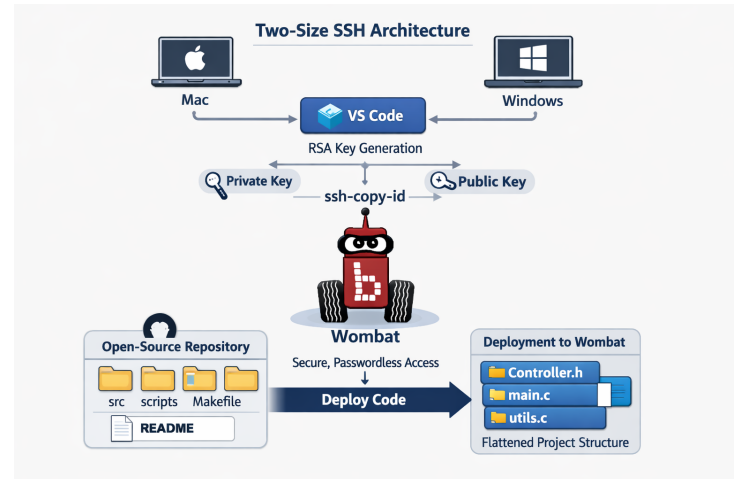


Fig 1: Visual representation of two sized fits architecture

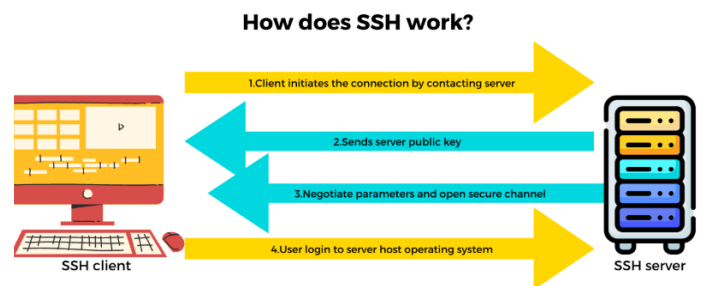


Fig 2: How an SSH works

I. INTRODUCTION AND BACKGROUND

The KIPR Wombat is an integral piece of the Botball competition, providing a platform for students to engage in autonomous robotics. In my experience in these 3 years of Botball, development on this platform has been confined to the KISS IDE, which prioritizes simplicity and practicality. However, as robotics tasks become more complex, the limitations of minimalistic editors become apparent. Professional developers rely on integrated development environments, or IDE's, that provide comprehensive code management, and bridging this gap is essential to prepare students for modern software engineering careers.

II. AN EASIER WAY TO CODE

The architecture I have developed shifts the primary development interface from the limited software of the Wombat to a high-performance and importantly local, coding interface. In this model, VS Code functions as the command and coding center, providing the developer with an environment optimized for coding and using AI tools like Ollama. Once code is written locally on VS, it is transmitted to the Wombat via a cryptographic network protocol used for operating network services securely over an unsecured network, which provides a robust, encrypted channel between the host workstation and the Wombat, called an SSH protocol (see Fig 2 for a visual representation of what an SSH is). This allows the students to utilize the superior interface of a modern interface while ensuring that the final build is

executed by the Wombat's internal compiler, maintaining perfect synchronization with the required hardware.

III. IMPLEMENTATION AND DEPLOYMENT WORKFLOW

A. *What is really happening?*

The implementation of this architecture is designed for a two size fits all approach, per se. (Visual representation of the following explanation is shown in Fig 2). Since the architecture syncs and compiles, at a glance, the same across multiple devices, and two sizes since there are both Mac and Windows compatibility. The initial configuration involves the generation of an RSA-formatted SSH key pair on VS code, which is on the local computer. By deploying these keys to the Wombat via `ssh-copy-id`, the user establishes a secure, passwordless authentication bridge that persists across sessions (meaning there is no need to keep redoing the setup, assuming it's the same device). This eliminates the need for repeated manual entry of the password, streamlining the development cycle significantly.

B. *The public repository*

Once the identity is established, file deployment is managed through a specialized integration within the VS Code application. I plan to open-source the repository that contains the necessary scripts and configuration files (namely the src file, the scripts, makefile, and readme [for easier use]), making this implementation accessible to the entire Botball community.

After taking the open-sourced github code, students can interact with it directly through the VS Code command palette, which acts as the primary interface for triggering deployment tasks. The deployment logic executes a transfer of local source files into the appropriate directory on the Wombat's file system, which can be changed from what is set on the open source repo; where the standard compilation process is then triggered.

C. Portability

To ensure the project is portable, the scripts automatically flatten the project structure, relocating all header and source files to singular directories. This necessitates that all header file inclusions remain flat, using syntax such as `#include "Controller.h"` rather than specifying directory paths, which ensures compatibility with the KIPR compilation environment while enabling local Git management for seamless version control and collaboration. KIPR natively uses this simplified structure, therefore making it necessary to port. You can find this written as notes in the README as well.

IV. ADVANTAGES

A. My experience and other common problems

The transition to a professional-grade editor yields immediate benefits in both code quality and velocity. Native Git integration enables students to manage complex codebases, experiment with features in branches, and track changes. My team, 0329, frequently overwrote each other's

code by accident, or were set back by weeks of progress due to a simple error that one person forgot to copy paste code from KISS IDE to our Git repository, or even more frequently forgetting we were connected to the bot's wifi when pasting code, leading to the code not being saved nonetheless. Furthermore, the extensible nature of VS Code provides access to specialized syntax highlighters that prevent common programming errors before the code is ever deployed to the robot. It also allows the integration of Large Language Models (LLMs), which assists in coding while being connected to the robot's wifi. This essentially eliminates the need to switch to the Internet - letting you code, deploy, make changes without syntax errors, and use LLMs to help code mundane sections.

V. PLACES TO IMPROVE

While the transition to a modern IDE offers substantial improvements, several areas remain for future optimization. The most immediate challenge is the compilation process. Relying solely on the Wombat's on-board hardware for compilation remains a factor that slows the process. Implementing a remote, cross-compilation model on the host's VS Code application could reduce the time it takes. The ability to have LLMs, code and deploy, all while being on the Wombat's wifi is what's necessary right now as an upgrade of sorts from the KISS IDE, but if compilation compatibility was developed with a port to VS Code it could mean that the host computer can be connected to regular wifi while coding the bot. Additionally, while the current "flattening" approach solves pathing

issues for header files, migrating toward a robust build system (like the one natively on VS code) would allow for more complex project hierarchies and easier to peruse code without compromising portability, though this would require a KISS IDE update. Finally, there is a significant opportunity to enhance the debugging experience. While this workflow currently manages code synchronization, integrating remote debugging software directly into VS Code would allow students to set breakpoints, monitor memory usage in real time, and in doing so move the process of coding a robot in the Botball robotics competition for both GCER and regionals a lot closer to a fully professional-grade robotics engineering environment.

V. AI IN BOTBALL

A central component of this modernization is the integration of local LLMs to facilitate an iterative development style. By running an LLM locally within the VS Code environment, students gain access to an intelligent assistant, which operates privately and locally on the student's machine. This model can be specifically prompted to understand the nuances of the Wombat's internal libraries, providing context-aware suggestions for sensor integration and motor control. The LLM-based development allows a conversational flow where the developer states the desired robotic behavior or specific outcome of the code, and the LLM translates these requirements into syntactically correct, hardware-aware code. This rapid, prompt-based iteration significantly lowers the time used to code

and allows for a more creative, experimental approach to problem-solving. The LLM that I am most familiar with porting to VS code is called Ollama (a gemma model specifically), and is the LLM optimized for in the code is this as well. AI is going to be a big part of the future, so this will help prepare students and lead them to a bright future.

VI. CONCLUSION

Moving away from restrictive, and often primitive editors in favor of a modern, SSH-based workflow represents a necessary evolution in robotics education. By embracing industry-standard tools like VS Code and augmenting them with local AI assistance, students can achieve a level of efficiency previously unavailable in the KIPR ecosystem. This not only solves the immediate technical challenges of development but also provides students, as many of us are, with the versatile skill sets required to excel in professional software development environments, and in doing so prepares them for the bright future ahead of them.

ACKNOWLEDGEMENTS

Thank you to parents for proofreading, and to Arush Garg for his guidance on this publication.

REFERENCES:

- [1] KISS Institute for Practical Robotics (KIPR), "Botball Botguy," kopr.org, 2026. [Online]. Available: kopr.org. Accessed: June 13, 2026.
- [2] Fokusy, "What is SSH and how does it work?," fokusy.org, 2026. [Online]. Available: <https://fokusy.org/what-is-ssh-and-how-does-it-work-an-easy-explanation/> Accessed: June 13, 2026.
- [3] Google Gemini, "AI-generated image of my Two Size architecture (botball image added on top)." Generated by Dhruv, June 13, 2026.
- [4] GS, Adarsh, "A Beginner's Guide to SSH: What it is and how to use it," blog.devops.dev, 2024. [Online]. Available: <https://blog.devops.dev/a-beginners-guide-to-ssh-what-it-is-and-how-to-use-it-27c118fec3d4>. Accessed: June 13, 2026.