# Determining Best Language to Implement Robot Swarm


## Abstract:

Programming languages all have their strengths and weaknesses, and different programs will be best implemented in some programming languages over another. Thus, it is important that companies all take careful consideration to which programming language is the best fit for their specific project. In this paper, we consider a project that requires a programming language that is portable and efficient. This is because the program will be run on a system with limited resources, so needs to be efficient with its usage of space and CPU. The company that is requesting the program also desires that the program be portable in case they want to switch hardware.

In this paper, we considered four programming languages: C++, Go, Python, and Java. We discussed the strengths and weaknesses of each one, and conclude that Go is the best programming to use. This is because although it isn't very well supported due to being a fairly new language, out of the four, Go is the most efficient and portable language.


## 1. Introduction and Requirements

Our goal for this project is to design a program for tiny flying robots that will kill insects. They will be used in areas that have high concentrations of flying insects, but where conventional chemical treatments aren't allowed. Each of these robots will be equipped with low-power CPUs and AI accelerators that will execute the program to recognize insects and utterly demolish their population to a little more than nothing. The goal of this paper is to decide which programming language would be best language to implement the proposed program.

Because of the restrictions on the hardware, we must choose a language that will be efficient with the limited resources, so that it can kill as many living creatures before having to recharge. In addition, the company requesting these autonomous killing machines would like for us to make the code portable, in case they would like to switch to a cheaper and cost-effective hardware vendor. This way they wouldn't be tied down to one vendor because of the programming language.

Furthermore, these robots will be deployed in swarms, so must be also able to communicate well with one another. Then finally, the language should be compatible with TensorFlow Lite and TensorRT, which are deep learning frameworks for embedded devices (similar to TensorFlow).

In this paper, we will consider four languages: C++, Go, Java and Python. We will compare the strengths and weaknesses of these languages in light of these requirements in order to properly conclude which language is most fit for this program.

## 2. C++

### 2.1 C++ pros:

C++ is a very popular language for embedded systems, and this is for good reason C++. C++ is good because it is very efficient with it's CPU usage. Although it's a more difficult language to pick up, the program can be optimized greatly and be very efficient. This is relevant for this project because we will be operating on hardware with low resources.

Additionally, C++ libraries for machine learning is very well supported, and has a sizable support base for TensorFlow Lite and TensorRT.

### 2.2 C++ cons:

On the other hand, because of the freedom given to C++ programmers, many problems arise when optimizing. Because memory has to be manually freed, programs can very easily run into problems with memory leakage due to an unfreed pointer. Additionally, race conditions may occur when trying to optimize a program by multithreading. This will be an issue because errors may arise much later, even after deployment of the robots, making the program unreliable.

Furthermore, not all C++ programs produce the same results on different machines. Some programs may not be compatible at all, or will give errors that don't manifest on another machine. In either case, additional testing is always required when switching to a different machine. Errors might be difficult to catch because they might only manifest in specific edge cases. Then if there are errors, the code will have to different on different machines to fix them. This would take more time to deploy, and would slow down the company. If the errors are not caught, then would prove to be a reliability issue.

## 3. Go (Golang)

Taking after C++, Go is a language created by Google developers which aims to combine, usability, efficient compilation and efficient execution of code.

**3.1 Golang pros:**

Because Golang is aimed to be a child of C++, it boasts the efficiency of it too. This again is relevant because of the low resource machines that are being used.

Golang's biggest advantage over C++ is that Golang has its own garbage collector like Java and Python. This means that programmers won't have to worry about memory leakage. This is important because this eliminates the possibility for many errors to even exist, thus increasing reliability greatly.

Additionally, Golang's concurrency mechanisms make it easier to write concurrent code. But this doesn't necessarily eliminate the possibilities of race conditions. So, programmers still must be careful. There is no huge advantage here.

Lastly, Golang is just a very easy programming language to use. It has a similar feel to C++ but has the friendly style that Python does. This is nice when debugging, because it's much easier to understand.

**3.2 Golang cons:**

The biggest downside to using Golang is that it's such a fairly new language, so it doesn't have much support yet. Despite having a few GitHub libraries to help with compatibility, Golang isn't as supported by TensorFlow as C++ or Python. This may prove to be a burden later when moving between machines.

Because similar to C++, Golang doesn't have a major advantage in computability. It will require more testing when switching machines. The good thing is that it won't require as much though, since there won't be memory leakage, and less problems will concurrency.

## 4. Java

**4.1 Java pros:**

The biggest advantage in using Java is that it is portable. Because of the Java Virtual Machine, it is almost trivial to move between machines. This is a huge advantage because it makes deployment very quick and easy.

Additionally, like Golang, Java has it's own garbage collector, so programmers will not have to debug for memory leakage. Multithreading is supported though, and while this is good because programs can be optimized greatly because of this, there still may be race conditions to check for.

**4.2 Java cons:**

The biggest strength of Java can also be its greatest weakness. If a machine doesn't have a Java Virtual Machine, a program can't run this program at all. This is a huge problem because it prevents a machine from even being considered at all, and will restrict the company that we work for.

Also like Golang, there is little support for Machine Learning. Since all libraries are in C++ or Python, this makes it difficult to implement the Machine Learning algorithms needed to create this robot swarm. In particular, TensorFlow Lite and TensorRT are both not greatly supported, which is a problem because this will be the library we will be using.

## 5. Python

**4.1 Python pros:**

The biggest advantage in using Python is its support. Python is one of the most popular languages to use for machine learning, thus most of the machine learning libraries are for Python. TensorFlow Lite and TensorRT are both well supported for Python, so it would be very easy to use.

Additionally, like Golang, Python is a very easy language to understand and read, so debugging would be easier.

**4.2 Python cons:**

One of the disadvantages of Python comes because it is not a compiled language. Since Python is a scripting language, the other compiled languages don't have to consider anything at runtime, and have a major advantage in speed.

Additionally, compared to C++, the portability is not much better. Java still wins in the portability race.

## 6. Conclusion

After considering the strengths and weaknesses of all of these languages, we can now conclude which of these languages is the best language for this specific project.

First we see that C++, although being the most efficient language, has too many chances for error since it has the largest risk for memory leakage and for race conditions that might only be found much later into deployment. Because of this, will have the most trouble being portable, since these edge case

error will have to be checked every time we switch machines. Thus it is not recommended that this program be coded in C++ because of these big drawbacks.

Next, Java is a clear winner in portability because of the Java Virtual Machine. But that advantage is completely nullified if the machine that we are using doesn't have a Java Virtual Machine in the first place. In addition to that, Java really isn't the first choice for machine learning because it isn't as well supported. Given these drawbacks, Java isn't recommended.

Next, we see that Python, although being friendly and extremely supported by the machine learning libraries, just isn't very efficient because it's not a compiled language like the others. Especially because we will be running this program on machines with limited resources, we need a language that makes very good use of the materials we have. So although Python isn't a bad language to use, it's not the best.

Lastly, Golang is the most balanced program all around. It is pretty efficient because it's a compiled language, is more portable because of the garbage collection, and is a very easy language to understand. The downside being that it's not very well supported at all because it's a fairly new language. But because of the advantages to using Golang over the other languages, it is recommended that Golang be used to code this program.

## 7. References


1. "Modernes C  ." *ModernesCpp.com*, www.modernescpp.com/index.php/race-condition-versus-data-race.
2. "The Myth That Python Is Slow." *Python For Engineers*, www.pythonforengineers.com/why-is-python-so-popular-if-its-so-darn-slow/.
3. "NVIDIA TensorRT." *NVIDIA Developer*, 23 Oct. 2019, developer.nvidia.com/tensorrt.
4. Roulo, Mark. "Java's Three Types of Portability." *JavaWorld*, JavaWorld, 1 May 1997, www.javaworld.com/article/2076944/java-s-three-types-of-portability.html.
5. "TensorFlow Lite." *TensorFlow*, www.tensorflow.org/lite.