BATS

By

Behrooz Badii

Aleksandr Borovinskiy

Tanya Shtemberg

Sui Sum Wong

Hey, it’s better than naming it STAB.

What's BATS and why did we make this language?

What's BATS?

BATS is…

- A geometric figure drawing language:
  - This language uses a series of Points to describe a line or geometric figure.
  - All figures can be represented as a series of Points that have been connected by a line (even a circle can be roughly represented using just a series of points)
- A language using a JAVA-like syntax and semantics.
- A simple and easy-to-understand language:
  - BATS has a specific purpose to it: drawing. The usage of Lines and Points, coupled with an already popular JAVA syntax makes BATS much easier to use and understand. However, with correct usage, this language can create both rudimentary and complex figures.

Why make BATS?

We made BATS because …

- A visual language is more appealing and fulfilling than a language with no visual aspect.
- Geometric figure drawing can produce some very beautiful and complex pieces of art.
- Previous geometric figure drawing languages have paved the way for us. We could make a stronger language learning from the work of previous groups, such as the highly acclaimed Mx language from last semester.
- After doodling in our notebooks during most of our first group meeting, in which we were supposed to figure out what language we wanted to create, the answer was right in front of us.
- Professor Edwards made us make a language, too.

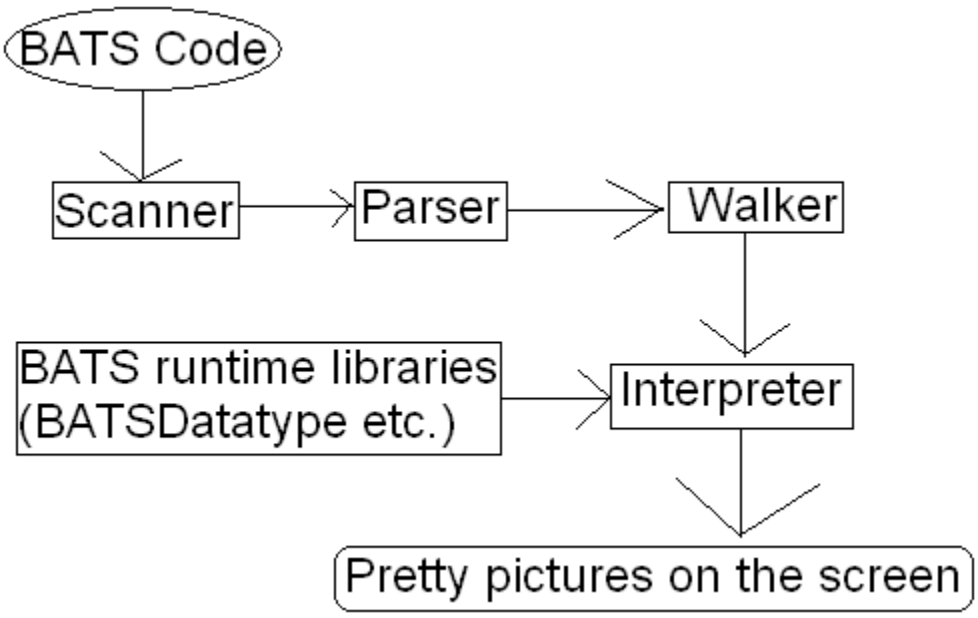
Features of BATS

BATS has…

- Block programming constructs like loops, conditionals, and functions.
- Variables.
- Lines, which are special arrays whose elements are Points.
- Points, which are a set of two Integer values depicting a spot on a graph.
- A special iterative for loop designed for Lines which does an action on every Point on the line.
- Built-in Functions:
  - Append Line to Point
  - Color Integer; Integer; Integer
  - Draw Line

How we implemented BATS

We used ANTLR to implement the Scanner, Parser, and Walker.

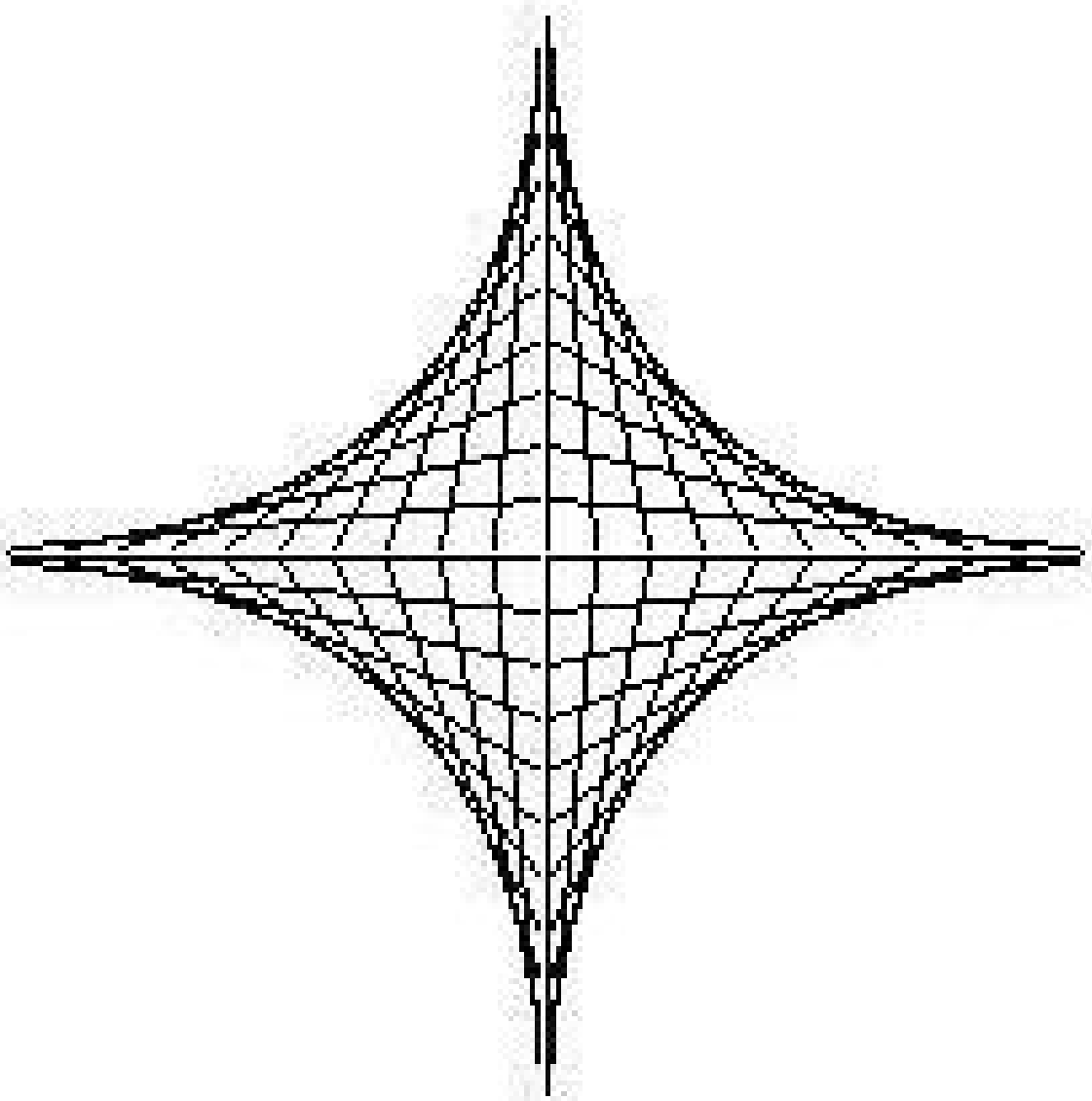


The Interpreter is JAVA code which takes in the values passed by the Walker (using the runtime libraries for static semantic analysis) and draws each Line in succession on a Java frame.

BATS: An Example

```
Global Line Xshape, Yshape, temp;
Global Point x, y;
Boolean Begin              #This is the main file of a BATS
                           #program, compiler seeks this out
                           #and starts running code from here
Int w, z;
Let w <- 100;
Let z <- 10;
Let x  <- ^100,0^;     #syntax of a Point is: ^ Integer, Integer ^
Let y  <- ^0,-100^;
Let Xshape <- {^ –100,0 ^ , x};#making x and y axes
Let Yshape<-  {y , ^ 0,100 ^};
Draw Xshape;                   #drawing x and y axes
Draw Yshape;
      While w > 0 Do
            Let temp <- {^ z,0 ^ , ^ 0,w ^ };
            Draw temp;         #draw top right quadrant
            Let temp <- {^ z,0 ^ , ^ 0,-w ^};
            Draw temp;         #draw bottom right quadrant
            Let temp <- {^ -z,0 ^ , ^ 0,w ^};
            Draw temp;         #draw top left quadrant
            Let temp <- {^ -z,0 ^ , ^ 0,-w ^};
            Draw temp;         #draw bottom left quadrant
            Let w <- w – 10;
            Let z <- z + 10;
      WhileEnd             #note, indentation is not required
Return True;
End                        #ends Begin function
```

The above example creates the following picture:



Further Work

In future versions of BATS, we would like…

- Better error handling.
- The ability to use libraries
  - We would like to include library files describing and depicting much-used polygons like rectangles and circles in our language.
- An actual polygon type.
- The ability to manipulate polygons:
  - Changing their color
  - Shrinking them
  - Increasing their size
  - Changing singular Points and their positions in polygons
- The ability to write text on the screen with different colors, fonts, and sizes.
- The ability to include actual java code in a special bracket or brace notation.
  - We feel that this could expand our language even if we do not or cannot implement certain items on our wish list.

Conclusion

To other groups, we recommend that they…

- Start early.
- Have a strong channel of communication. The speed at which our inboxes were growing was unbelievable in the last few weeks.
- Have faith in your group members, for they will always pull through.
- Should not be afraid to ask questions from teaching assistants and the professor.
- Be open to ideas from all group members. Even though the team is recommended to be a dictatorship, recommendations and changes should be welcomed.
- Work together. The more often the group works together, the less discrepancies there are.

With our language, we hope to…

- Expand our language.
- Make our language highly accessible.
- Modify the language to make it even easier to use.
- Serve as an example for future groups.
- ~~Bring peace to the world.~~
- ~~End all wars.~~
- ~~Discover time travel.~~
- ~~Win the Turing Award.~~
- ~~Publish a paper in a prestigious journal.~~
- ~~Become filthy rich and famous.~~
- ~~Do well at Columbia University~~
- ~~Do well in this class.~~
- Do well on the final project.