Roomba!

Goal: whole-environment coverage

Roomba

Goal: whole-environment coverage with only ***local sensing***…

Challenges:

- Can only sense it's immediate surroundings – doesn't see global picture
- Can't tell "vacuumed" from "unvacuumed" area – no memory

Roomba is a computer!

- We will use a special programming language that can instruct a computer similar to Roomba.
- This programming language contains instructions with the aforementioned challenges in mind.
- We call this special computer "Picobot."

Another language? Really?

Python

General-purpose language

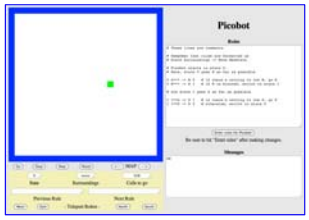
you might see 50% by the end of the semester

Picobot

Special-purpose language

you'll see 100% by the end Of the lecture

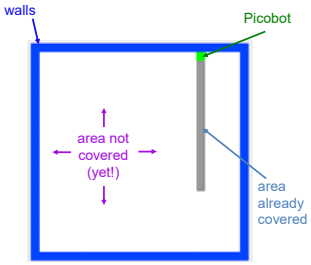
Picobot!



The Picobot simulator

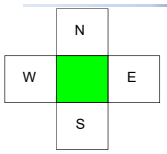
http://www.eg.bucknell.edu/~csci203/common-files/course-docs/picobot/picobot.html

Picobot!



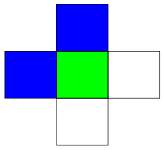
Goal: whole-environment coverage with only ***local sensing***…

Surroundings



Picobot can only sense things directly to the N, E, W, and S

For example, here the surroundings are



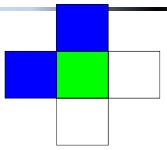
NxWx

N E W S

Surroundings are always in NEWS order.

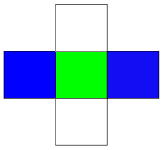
What are these surroundings?

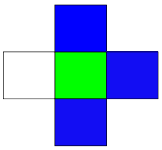
Surroundings are always in NEWS order.



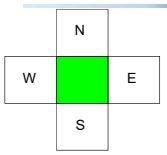
N E W S

NxWx



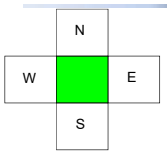


Surroundings



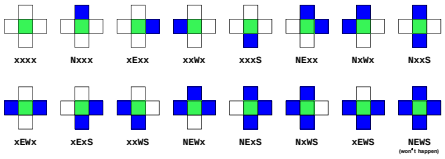
How many distinct surroundings are there? E.g., the one to the left would be described as "xxxx" or "all open."

Surroundings

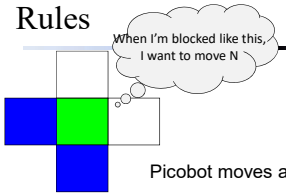


How many distinct surroundings are there?

2^4 == 16 possible …



Rules



I should move N.

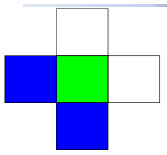
Picobot moves according to a set of rules:

surroundings	direction
xx**WS**	→ N

If I see xx**WS**, *Then I move* **N***orth*

Note: X in direction means don't move.

Wildcards



Asterisks * are wild cards. They match walls ***or*** empty space:

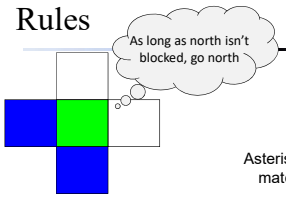
surroundings	direction
x***	→ N

N must be empty

EWS may be wall ***or*** empty space

Wild stars? You should visit Alpha Centauri!

Rules



I should move N.

Asterisks * are wild cards. They match walls ***or*** empty space:

surroundings	direction
x***	→ N

If I see North is free (no matter what other walls there are)

Then I move **N***orth*

Note: X in direction means don't move.

Model

Picobot checks all of its rules.

If it finds a matching rule, that rule runs.

Only one rule is allowed per state and surroundings.

Not allowed

surroundings		*direction*
*x****	->	*N*
*x****	->	*X*

What will this set of rules do to Picobot?

surroundings		direction
x***	->	N
N***	->	X

How can we get back down the screen?

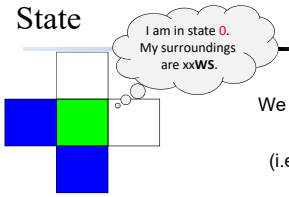
State in picobot

State describes what it will do.
Picobot can have many rules (you decide!).

"Moving East until I can't"

All computers have state:
The internal context of its computation.
It represents what the computer is doing.

State



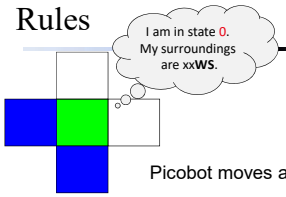
We use a number to represent the state of picobot.

(i.e., picobot has memory to store a single number)

Picobot always starts in state 0.

State and **`surroundings`** represent everything Picobot knows about the world

Rules



Aha!
I should move N.
I should enter state 0.

Picobot moves according to a set of rules:

state	surroundings		direction	new state
0	xxWS	→	N	0

If I'm in state 0 *seeing* xx**WS**,

Then I move **N***orth, and "change" to state* 0.

Note: X in direction means don't move.

Model

Picobot checks all of its rules.

If it finds a matching rule, that rule runs.

Only one rule is allowed per state and surroundings.

What will Picobot do following this set of rules?

Try it out!

state	surroundings		direction	new state
0	x***	->	N	0
0	N***	->	X	0

How can we get back down the screen?

Model

Picobot checks all of its rules.

If it finds a matching rule, that rule runs.

Only one rule is allowed per state and surroundings.

Ping-Pong Picobot Rules

state	surroundings		direction	new state
0	**x*****	->	N	0
0	**N*****	->	X	**1**
1	*****x**	->	S	1
1	*****S**	->	X	0

To do

Write rules that will always cover this room.

hw0, Problem #3

your rules should work *regardless* of Picobot's starting location

To do

Write rules that will always cover this room.

hw0, Problem #4

your rules should work *regardless* of Picobot's starting location

Tips for Picobot problems

- Thinking about the CS questions before diving into the programming will help!
  - Imagine you're blindfolded in the room. How would you solve it?
  - Solve it FIRST in English, then try to figure out the algorithm (don't worry about code!).
  - For each sentence in English, that might be a different state.
  - If you find that rules conflict with each other, you might need a different state.

CS ~ *complexity* science

Information is intrinsic to every system…

How can we *benefit from* this information?

"construct with "

Representing ***it*** efficiently? … Transforming ***it*** effectively? … Measuring ***it*** possibly?

How might we ***measure*** these rooms' **complexity**?

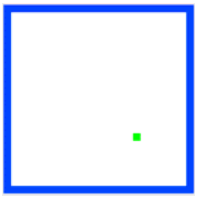
CS ~ *complexity* science

Information is intrinsic to every system…

How can we *benefit from* this information?

"construct with "

Representing ***it*** efficiently? … Transforming ***it*** effectively? … Measuring ***it*** possibly?



our best: 3 states, 6 rules

How might we ***measure*** these rooms' complexity?

How many states and rules are necessary ?

How much information does each room contain ?

our best: 4 states, 8 rules

CS ~ *complexity* science

Information is intrinsic to every system…

How can we *benefit from* this information?

"construct with "

Representing ***it*** efficiently? … Transforming ***it*** effectively? … Measuring ***it*** possibly?

As a file: ~5000 bytes

How might we ***measure*** these rooms' complexity?

How many states and rules are necessary ?

How much ***information*** does each room contain ?

As a file: ~20,000 bytes!