Creating "Journey" In AgentCubes

Student Version – No AgentCubes Experience

You are a traveler on a journey to find a treasure. You travel on the ground amid walls, chased by one or more chasers. The chasers at first move randomly on the ground, and later, begin to chase by following your scent. When you collect the treasure, you win. If a chaser catches you, you lose.

Created by: Cathy Brand, University of Colorado, School of Education

This curriculum has been designed as part of the Scalable Games Design project.
It was created using portions of prior work completed by Susan Miller.

This material is based upon work supported by the National Science Foundation under Grant No. DRL-1312129 and CNS-1138526. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Student Handout 1A: No Prior Experience with AC

Part I - Basic Game

Initial Story: You are a traveler on a journey to win a treasure. You travel on the ground amid walls along with one or more chasers. The chasers move randomly on the ground. When you collect the treasure, you win. If a Chaser moves next to you, you lose.

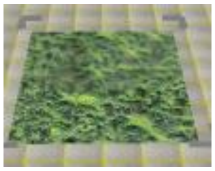
Create these Agents:

Choose the Inflatable Icon – People category and pick 2 images for your agents:

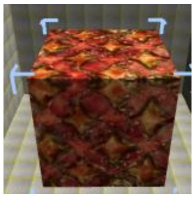
Traveler

Chaser

Choose a Tile to be the ground. Pick a color that contrasts with your traveler and chaser.

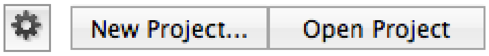
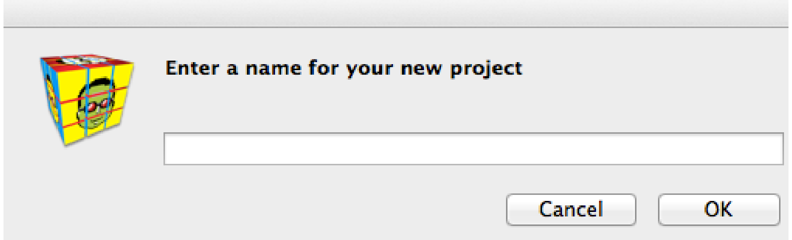
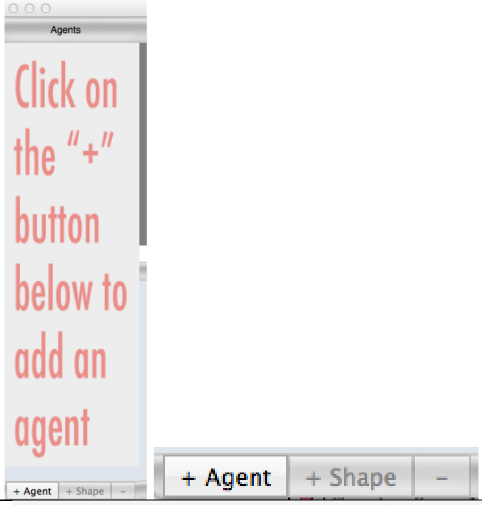
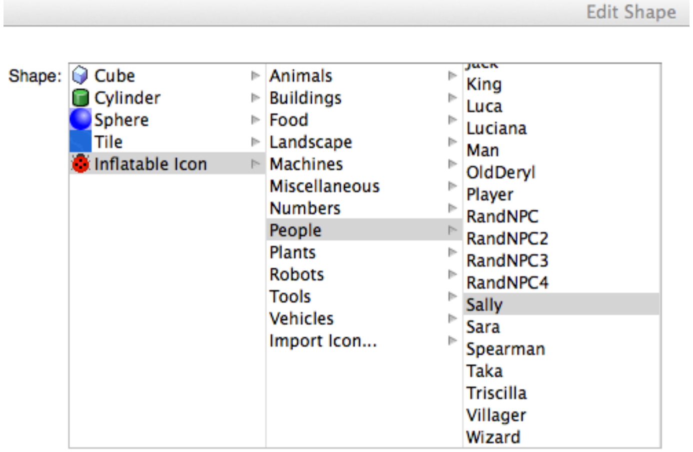


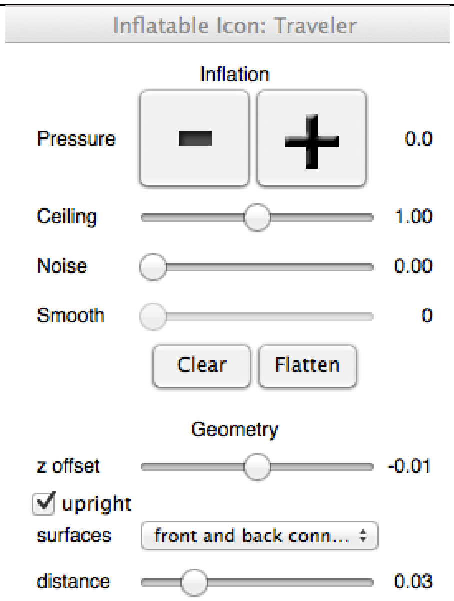
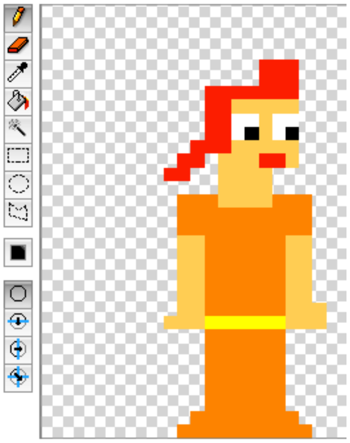
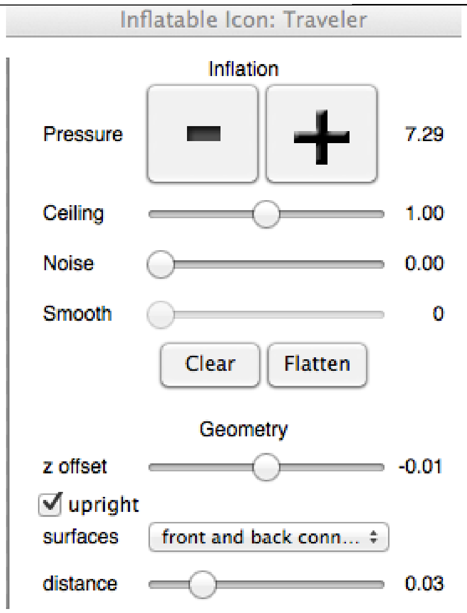
Choose a Cube for making walls. Pick a color that contrasts with the ground.



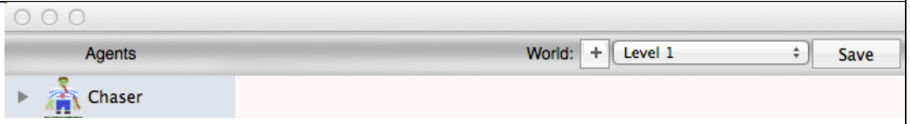
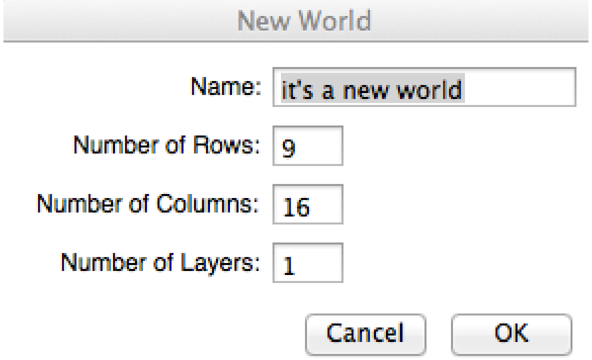
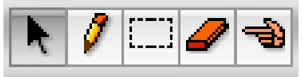
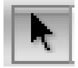
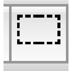
Choose an image for your Treasure agent from the Inflatable icon – miscellaneous category.

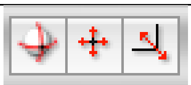
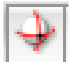
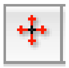
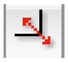
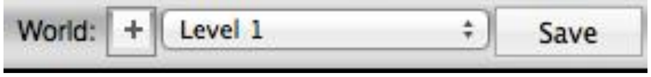
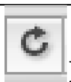
DRAFT **3-D Journey (Continued)**

Step 1	**Create Game** **Click on the New Project button at the bottom of the Project Chooser Window.** **Or go to the AgentCubes File menu and click on New Project.**	
Step 2	**Name the Game** **Name it Journey and click OK.**	 Enter a name for your new project
Step 3	**Create Agent** **Click on +Agent button in the lower left corner of the AC window.**	 Click on the "+" button below to add an agent
Step 4	**Choose *Inflatable Icon* from the Shape list on the left, then *People* from the center list and finally *Sally or another name* from the list on the right.** **Click OK to save your agent.**	 Edit Shape Shape: Cube Cylinder Sphere Tile Inflatable Icon Animals Buildings Food Landscape Machines Miscellaneous Numbers People Plants Robots Tools Vehicles Import Icon... Jack King Luca Luciana Man OldDeryl Player RandNPC RandNPC2 RandNPC3 RandNPC4 Sally Sara Spearman Taka Triscilla Villager Wizard

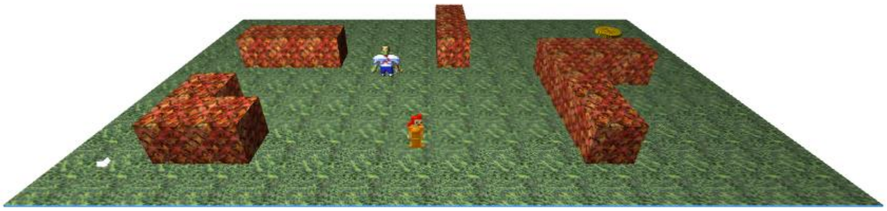
Step 5	**Draw your own traveler by clicking on the picture of a person in the list of agents.** **Use the clear button to erase the person.** **Or just use the pencil tool to add details to the person.**	 Inflatable Icon: Traveler Inflation Pressure 0.0 Ceiling 1.00 Noise 0.00 Smooth 0 Geometry z offset -0.01 ☑ upright surfaces front and back conn... distance 0.03
Step 6	**Click on Pencil tool for drawing.** **Click on color well to choose a color.**	
Step 7	**Use the + button to inflate your drawing and make it 3D.** **Check the *upright* box to stand your Traveler up.** **Pick *front and back connected* next to surfaces and then move the *distance* slider to make a wider 3D image.**	 Inflatable Icon: Traveler Inflation Pressure 7.29 Ceiling 1.00 Noise 0.00 Smooth 0 Geometry z offset -0.01 ☑ upright surfaces front and back conn... distance 0.03

DRAFT 3-D Journey (Continued)

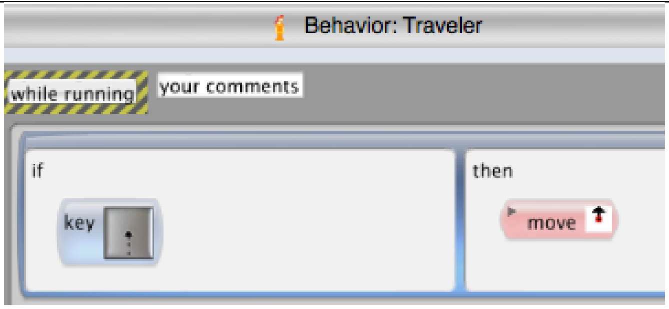
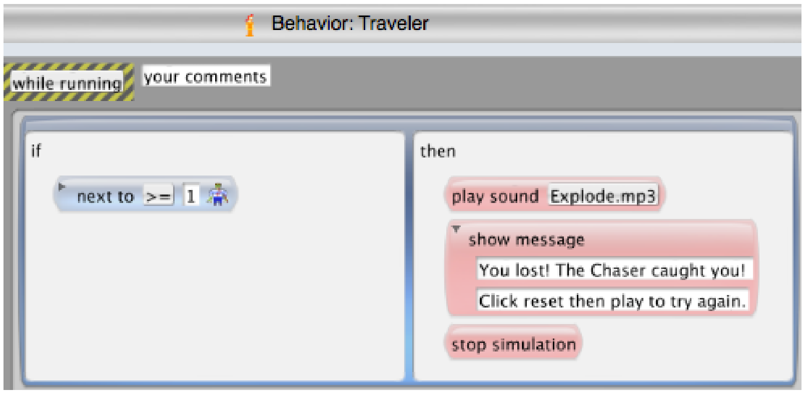
Step 8	**Create Wall, Ground, Chaser and Treasure agents.** **Pick a Cube for the Wall and a Tile for the Ground. Pick inflatable agents for the Chaser and the Treasure.**	
Step 9	**Create a world** **Click the + next to World in the top bar of the AgentCubes window.**	
Step 10	**Name Your New World "Level 1".** **Do not change the numbers for your first world.** **Click OK**	New World Name: it's a new world Number of Rows: 9 Number of Columns: 16 Number of Layers: 1
Step 11	**Try out the World tools** **Select tool for moving agents** **Pencil tool for drawing agents on World** **Tile tool for drawing groups of agents** **Eraser tool** **Trigger tool that calls a user-defined method** **Use the tools to design your Level 1 World.**	   **Do not place a Ground agent over another agent. This means if you place a Chaser on the World, do not draw the Ground over it without erasing the Chaser first.** **Cover the World with a layer of Ground Tiles. Put a few Walls**

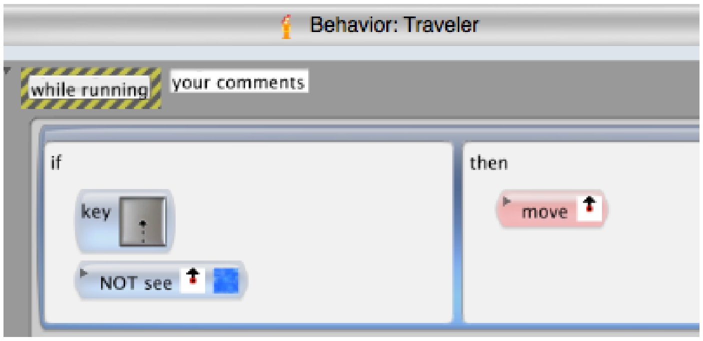
		on it. Place your Traveler, 1 Chaser and a Treasure on your World.
Ste p 12	**Try out the 3D movement tools** **Rotate** **Pan** **Zoom** **Move your World so it looks 3D.**	   
Ste p 13	**Use the save button to save your World when you like the way it looks!**	 **Reset button** **Save your World when it is set up to start a game. Then use the Reset button to return to the saved starting point for your game.**

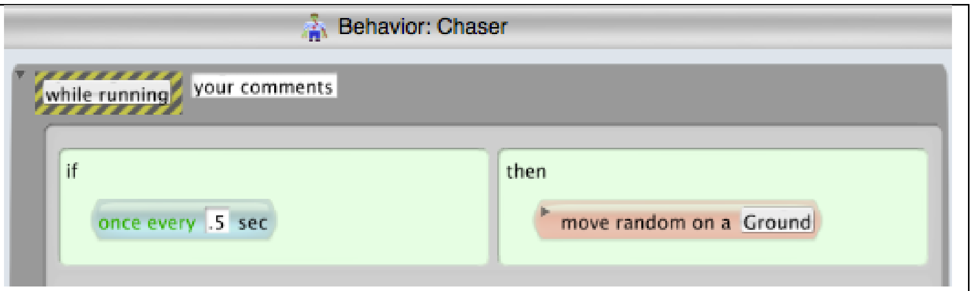
Your Level 1 World might look like this:



Step 14	**Create behaviors for your Traveler.** **Click on the Traveler to see its behavior.**	We will use rules to give behaviors to our agents. Rules are made up of an IF-THEN statement. In order to control the Traveler's movements with the cursor keys, we need this rule "IF the Up key is hit, THEN the Traveler will move up." The Traveler needs 3 more movement rules so there is a rule for each direction (Up, Down, Left, Right).

Step 15	**Create a behavior.** **The Traveler's rule will be blank. You are going to drop and drag the conditions (on the left) and the actions (on the right) to create the rules.**	 Behavior: Traveler while running your comments if key then move **Take a look at this rule…it says,** **IF I <u>click on the up arrow</u>, THEN <u>my traveler will move UP</u>** **Create the rules to have the traveler move up, right, left and down.** **NOTE: Each rule has to be separate…use the +Rule button to create each new rule.**
Step 16	**Make a rule so the Traveler loses when the Chaser comes near.**	The Traveler must lose the game if one or more Chasers catch her. How do we write 1 or more mathematically? **Use " >= 1".**  Behavior: Traveler while running your comments if next to >= 1 then play sound Explode.mp3 show message stop simulation You lost! The Chaser caught you! Click reset then play to try again. **Use the *reload World* or *stop simulation* actions to end the game!**
Step 17	**Make a rule so the Traveler wins when s/he collects (erases) the Treasure.** Use the **reload World** action to take the World back to its saved state and let the player go right on playing.	

	Use the **stop simulation** action to stop the game. Then the player must click on the reload button ↻ on the top bar of the AgentCubes window to return the game to its saved state and then click on the green triangle to play again.	
Step 18	**Prevent your Traveler from walking through walls.** **Part a) Add the code shown.** **Part b) Add code for the remaining directions.**	Work with the person next to you to figure out how to prevent the Traveler from walking into a wall. Here is one way to think about it.  Behavior: Traveler while running your comments if key NOT see then move **Note an important programming point**: The two conditions are in the same box…this is an AND statement. It reads like this: *IF the up arrow is pressed AND the traveler does NOT see a wall above him* *THEN he moves up*
Step 19	**Arrange your Traveler's rules so that rules which do special or unusual things appear first, followed by rules which control more common behavior.**	**What is the most important thing the Traveler does?** Win or lose a game! Put these rules first by dragging them to the top of the Traveler's while running method! **What does the Traveler do most of the time when you are playing the game?** Move around the World. Drag these rules below the Win and Lose rules.

Step 20	**Program the *<u>Chaser</u>* to move randomly by putting this rule in the Chaser's rules.**	 Behavior: Chaser while running your comments if once every .5 sec then move random on a Ground ***Click on the agent to add behaviors to that agent***

Student Handout 2
Part 2 – Making the Chaser Chase the Traveler

Step 1:

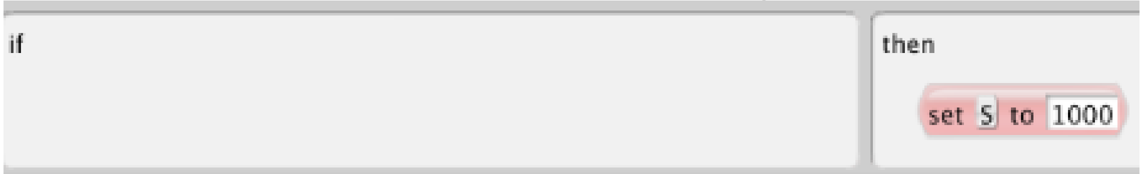
So far, your Chaser just moves randomly…he doesn't actually chase the traveler, does he? That's about to change!

We will make the Chaser pursue the Traveler agent using a search algorithm called "hill climbing." An algorithm is a set of rules followed by an agent to achieve a goal. Imagine the traveler agent emits a scent. Using the hill climbing search algorithm, the Chaser finds the direction in which the scent is strongest and moves that direction, following the Traveler.

The scent will spread out or be **propagated** by the ground agents using a computational thinking pattern called "diffusion" which copies the physical process of diffusion by which molecules move from areas of highest concentration to areas of lowest concentration. In this game, the values spread out from the Traveler to all the ground agents in the World. The values are highest in the ground agents close to the Traveler and smallest in the ground agents far away from the Traveler.

We will introduce the concept of an "**agent attribute**," which is a piece of information that is stored within each occurrence of an agent. Computer scientists call this attribute a **local variable** because each agent has its own copy of it and each copy has its own value.

First, let's make sure our traveler gives off a scent. To do this, we need to set the value of an attribute named "S" which stands for Scent. The S attribute is associated with the Traveler and is set in the Traveler's **last (lowest) rule in the while running method box**.



if

then

set S to 1000

This rule says to the Traveler, "Always give yourself a scent at level 1000." In other words, if no rule above it has a true condition, this rule's condition will evaluate to true and the Traveler's local variable S (Scent) will be set to 1000.

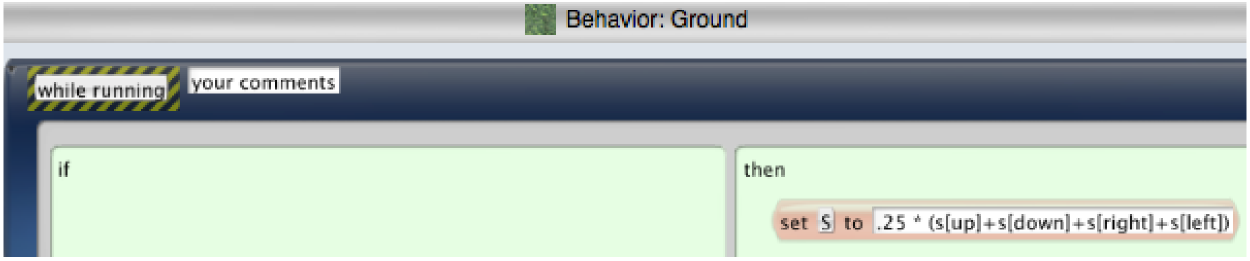
This rule should be AFTER all the other rules for the Traveler, at the end of the list so that the Traveler wins, loses or moves before setting S to 1000 because those actions are more important to the game.

Step 2:
Now, since the scent is diffusing, or spreading out from the Traveler, we need to find the value of the scent in each ground agent. Imagine that the smells are coming in from the North, South, East and West of each ground agent. The value of the smell in any ground agent, then, is the average of the smells in the four surrounding agents. How will you program that?

The ground agent will have the behavior below; the single action is to calculate and store the average of the values of the four surrounding agents' S attributes. Remember, you used the arbitrary name of the agent attribute "s" (for scent).

The "set" action sets each ground agent's attribute "s" to the **average** of the values of the attributes in the agents above, below, and on each side:

s = 0.25*(s[up]+s[down]+s[right]+s[left])



Behavior: Ground

while running | your comments

if

then

set S to .25 * (s[up]+s[down]+s[right]+s[left])

Helpful Tips

Match both the parentheses "(" and the brackets "[" as shown in the equation.

Why do we multiply by 0.25?

When you find the average of a set of numbers, you add them up and divide by the number of numbers.

In this case, dividing by 4 is the same as multiplying by 0.25

A METHOD is a set of rules to follow in a specific situation. You can create a METHOD by clicking the +Method button at the bottom of the AgentCubes window.

Step 3:

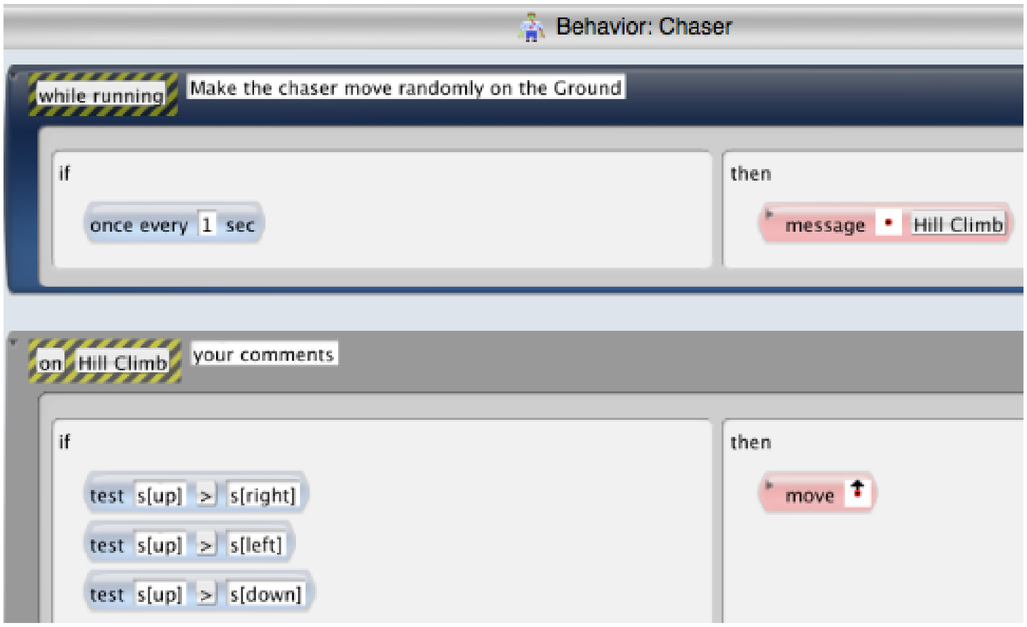
For the Chaser to know which way to walk, it has to determine where the scent is the strongest. If this were real life, it would smell up, smell down, smell left and smell right. Wherever the smell was strongest, it would walk in that direction. We need to program the Chaser to do this.

We will create a METHOD that enables the Chaser to do a hill climbing search so that it moves in whichever direction the scent is strongest.

Take a look at the programming below. Here's what it says…

The rule in the while running method says:
ONCE EVERY 0.5 seconds, send me a message to do my Hill Climb method.

The first rule in the Hill Climb method says:
IF the smell **above you** is greater than or equal to any of the other smells in different directions (down, left or right), THEN move up.



Behavior: Chaser

while running — Make the chaser move randomly on the Ground

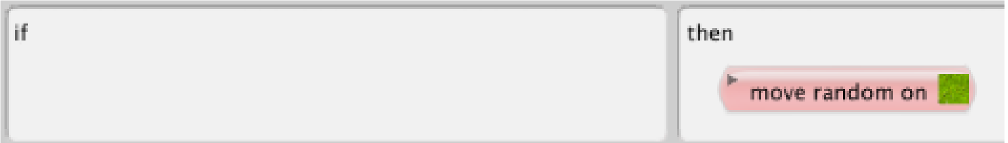
if
once every 1 sec
then
message • Hill Climb

on Hill Climb — your comments

if
test s[up] > s[right]
test s[up] > s[left]
test s[up] > s[down]
then
move ↑

Now, add the rest of the rules so that the Chaser knows what to do if the smell down (s[down]) is greater…What if the smell to the left is greater? What about the smell to the right?

Add this 5th rule at the **bottom** of the **hill climbing method box**:



if
then
move random on 🟩

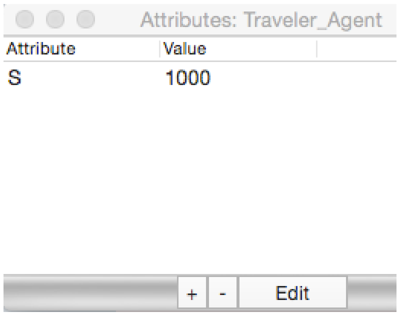
With this rule added, the Chaser will always make a random move if the S values in all 4 directions are the same. For example, the S values may all be equal to 0 if the Traveler's S value has not diffused all the way across the world yet.
Note: this rule must be the last rule in the Chaser's hill climbing method!

Student Handout:

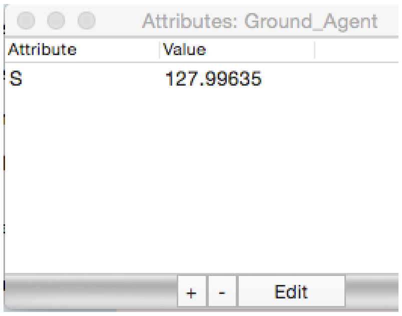
Troubleshooting Guide for Journey Part II

Diffusion and Hill Climbing

To determine what is happening in your game, it is sometimes helpful to look at the agent attributes. Reset your game, then click run followed by a click on stop. Do not reset at this point. Since your game has run briefly, your Traveler now has a scent. You can see his diffused scent (the value of s) by clicking on the ground agents with the big arrow tool and then going to the AgentCubes Window menu and selecting the **Show Agent Attributes** option. The attributes window will appear with the value of S for the agent that you clicked on.

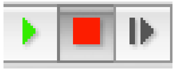

Attributes: Traveler_Agent

Attribute	Value
S	1000


Attributes: Ground_Agent

Attribute	Value
S	127.99635

Try checking the attributes of the four ground agents around the Chaser (up, down, left and right) and then single stepping the game using the black triangle next to the stop button Does your Chaser moves in the direction you expected him to go?



If your game isn't working, it's time to do some troubleshooting.
Check the following:

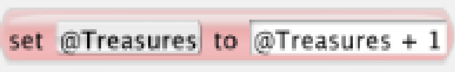
- The Traveler's "set S to 1000" rule must be the last rule in the while running method.
- In the Chaser's rules, the method name must be the same in the message action and the black and yellow striped method name tag on the hill climbing method!
- Use of parentheses "(" and brackets "[" in the Ground agent rule must be correct. Look at the picture of the Ground agent's equation 2 pages ago and compare it to the equation in your ground agent.
- Check your hill climbing rules again and make sure that the arrows in the actions point the correct direction and that the conditions for each rule are correct

Student Handout 3
Part 3:
Adding Challenge to the Game – Polling and Broadcast

We will add another challenge to our Journey Game. Now the Traveler must "collect" – that is move on top of – multiple treasures in order to win. The game does not end until all of the treasures are collected.

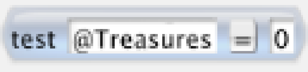
To accomplish this, we introduce the concept of SIMULATION PROPERTIES, which are named values that can be used and checked by all agents in a project.

A Simulation Property name is always preceded by an "@" when it appears in an action



set @Treasures to @Treasures + 1

or a condition



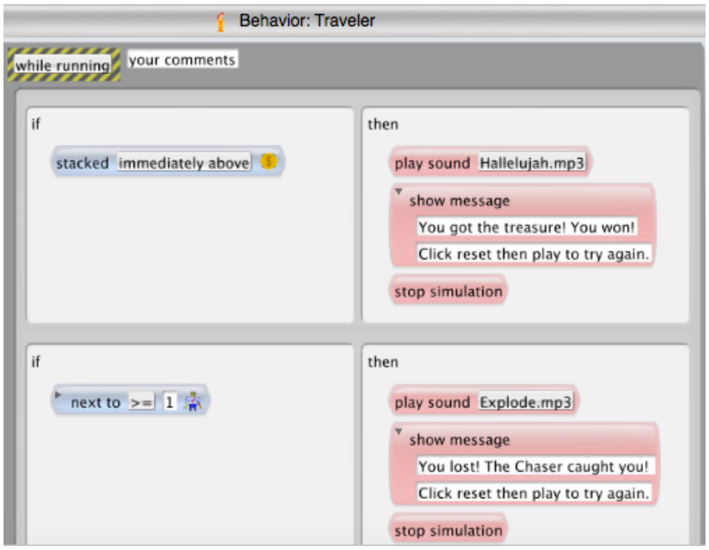
test @Treasures = 0

.

The @ differentiates simulation properties from agent attributes.

We will create a new agent, the "Controller" to manage the process of polling the treasure agents to determine when they are all "collected"; that is, when there are none left on the worksheet. To begin, we must change the behavior of the traveler agent so that it no longer declares the game is over when it moves on top of the treasure.

Step 1: Remove the win rule from the Traveler that makes the game end when she moves above the treasure.



Behavior: Traveler

while running | your comments

if
stacked immediately above
then
play sound Hallelujah.mp3
show message
You got the treasure! You won!
Click reset then play to try again.
stop simulation

if
next to >= 1
then
play sound Explode.mp3
show message
You lost! The Chaser caught you!
Click reset then play to try again.
stop simulation

Highlight the rule by clicking on the bar between the condition and action. Then press the delete button on your keyboard.

DRAFT 3-D Journey (Continued)

Step 2: Create a Controller agent
Create a Controller agent, using a colorful tile or any predefined shape or drawing your own. Use the pencil tool to place **one** Controller on your World.

Step 3: Counting up the treasures to see if you won

Imagine this conversation…

The teacher has given an assignment to the class and wants to know if everyone is finished. She says to the class, "Put your hand up if you are still working." Hands go up. She counts them – there are five students still working. "Okay, put your hands down and keep working."

A few minutes later, she does it again. She says to the class, "Put your hand up if you are still working." Hands go up. She counts them – there are two students still working. "Okay, put your hands down and keep working."

A few minutes later, she does it again. She says to the class, "Put your hand up if you are still working." This time, no hands go up. "Everyone is done, put your books away."

That process is similar to the way polling will work in your program.

Once per second, the Controller will say, "Treasure agent count starts at zero" (like the classroom, no hands are up when the teacher asks who is still working).

When the treasure agents 'hear' the Controller ask (broadcast) the question, the treasure agents respond back (raise their hands).

The controller checks the treasure agent count. If this count is more than zero, nothing happens and the game continues. If the answer is zero (meaning that there are no remaining treasures on the board), the game ends.

Definition: Computer scientists call the process of making a decision by sending a message to multiple recipients and checking responses **polling**.

How does Polling work?

In its "While Running" method, the Controller agent first sets the simulation property "@Treasures" to zero.

Then it broadcasts a signal "Count" to all treasure agents. Each treasure agent responds by adding one to the @Treasures simulation property.

Finally, the Controller calls upon the "Check Win" method. The player wins if no treasure agents are left in the world, which is determined by the @Treasures simulation property being zero.

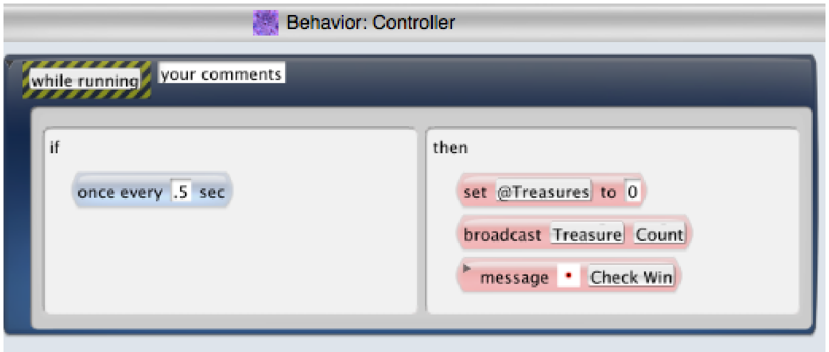
Create the rule in the while running method of the Controller:

Pick a time interval for the **once every** condition

Add these three actions to the same rule:

1. Set the value of the simulation property Treasures to zero. (this is like the teacher saying “hands down”). **Note you must write @Treasures in the set action!**
2. Use broadcast to ask all the treasure agents to evaluate the rules in their Count methods if they are still on the World.
3. Send me (the Controller) a message to evaluate the rules in my Check Win method to see if the Traveler has collected all the treasures and won the game.

The single rule in the Controller’s **while running** method should look like this:



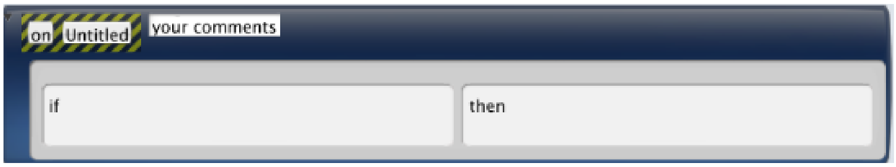
Behavior: Controller

while running | your comments

if: once every .5 sec

then:
- set @Treasures to 0
- broadcast Treasure Count
- message • Check Win

Click on the +Method button below the Controller’s rules. This method box will appear in the Controller:

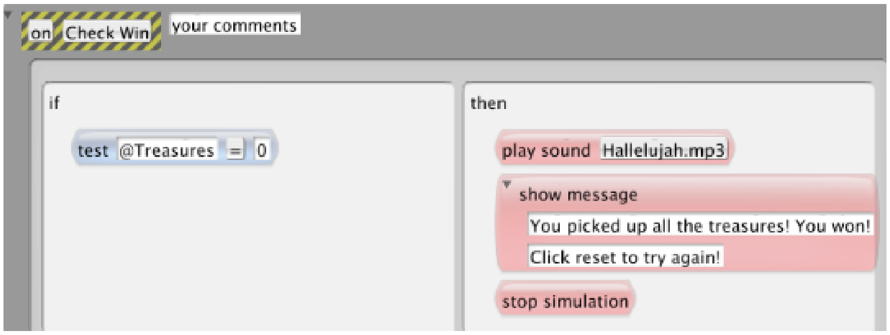


on Untitled | your comments

if

then

Click on the word “Untitled” in the upper left corner and choose the same name that you entered the message action from the rule in the Controller’s **while running** method.



on Check Win | your comments

if: test @Treasures = 0

then:
- play sound Hallelujah.mp3
- show message
  You picked up all the treasures! You won!
  Click reset to try again!
- stop simulation

Make the rule for the Check Win method:

1. Drag in the **test** action and use it to check whether the value of the simulation property Treasures equals zero because all the treasure agents have been collected by the Traveler. **Note you must write @Treasures in the test action!**
2. If the treasure agents are all gone, do the win actions. Remember to **stop** the simulation or **reload** the World to end the game!

Treasure behavior changes: There are two behavior changes required for the treasure agent.

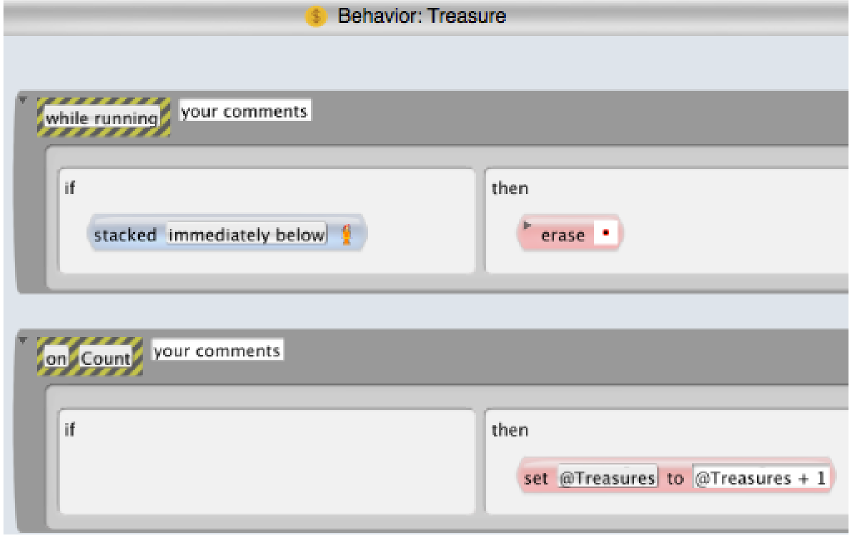
- The first step is to have the treasure be collected by the traveler. We can simulate this by **erasing** the treasure agent when the Traveler moves on top of it.

- The second behavior change for the treasure agent is to respond to the Controller's broadcast by evaluating the rule in its **Count** method, which updates the Treasures simulation property.

This second change is in the form of a separate method; it is not part of the continually running "While Running" method, since it only runs when called by the controller agent.

During **Count**, each remaining treasure agent will increase (or **increment**) the value of the Treasures simulation property. If no treasure agents remain in the World, then the value of the Treasures property will be zero, which the controller agent will detect and declare the game won.

Here is the behavior for the Treasure Agent:



Behavior: Treasure

while running — your comments
- if: stacked immediately below
- then: erase •

on Count — your comments
- if:
- then: set @Treasures to @Treasures + 1

The rule in the while running method erases the Treasure agent when a Traveler is on top of it.

The action in the Count method adds one to the value of the Treasures simulation property. **Note you must write @Treasures in the set action!**

Before you test this, check your world. In the picture above, we have placed **one** Controller tile in the right corner of the World.

In addition, we have placed several additional treasures on the worksheet, so that the traveler must "collect" – that is, move on top of each of them in order to win the game.

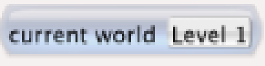
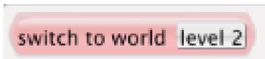
Adding Levels to Your Game:

Now that you have made your project more like a real game with Chasers that really chase the Traveler and multiple treasures that must be collected to win, it is fun to make several Worlds so that your player can try to win multiple levels.

How can you make one World more challenging than another?

1. Thank about the arrangement of the Walls. Is it easier or harder for the Traveler to escape the Chasers in a more open maze with fewer walls?
2. Think about the number of Treasures. Is it easier or harder for the Traveler to win when s/he must collect a larger number of Treasures?
3. Think about the number of Chasers. What number of Chasers would make it harder but not absolutely impossible for your Traveler to win?

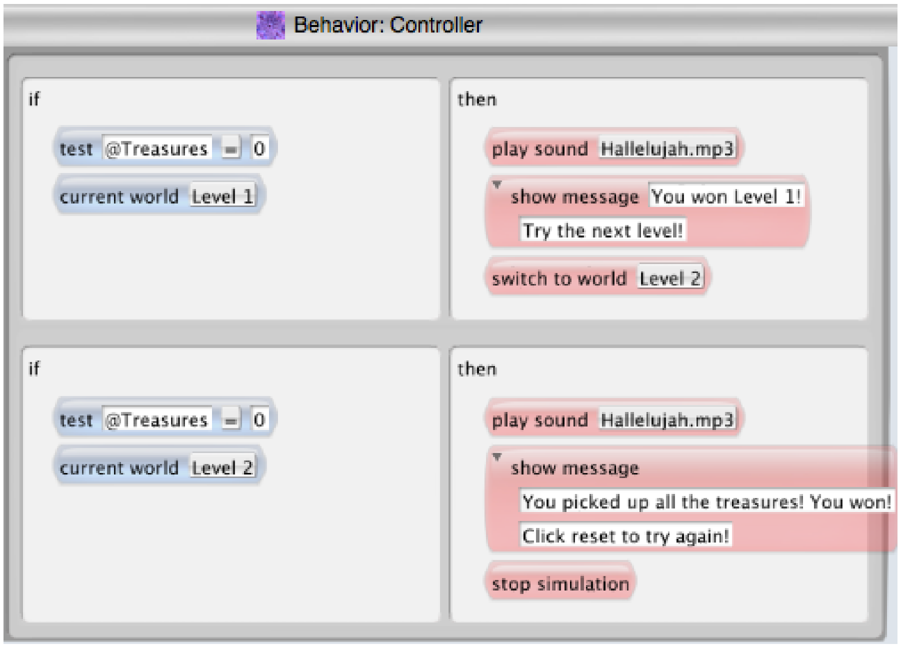
How do you make your Traveler move automatically from one level to another?

- AgentCubes has a condition that checks which World the agent is in right now  current world Level 1 and an action that lets the agent switch Worlds  switch to world level 2 . The example Worlds were named "Level 1" and "Level 2" when they were created.

Where would you put rules that use these actions?

- Think about when the player should switch levels: not in the middle of exploring a World but after winning a particular level.

Go to the Controller and replace the single win rule in Check Win with these 2 rules (or more if you have more than two Worlds):



Behavior: Controller

if
test @Treasures = 0
current world Level 1

then
play sound Hallelujah.mp3
show message You won Level 1! Try the next level!
switch to world Level 2

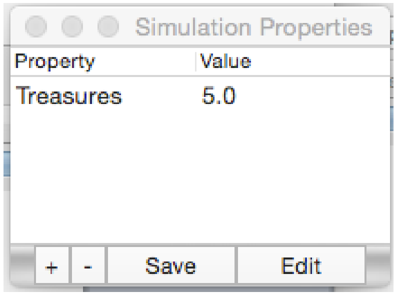
if
test @Treasures = 0
current world Level 2

then
play sound Hallelujah.mp3
show message You picked up all the treasures! You won! Click reset to try again!
stop simulation

Test your Worlds on your friends! How many Levels can they win?

Student Handout:
Troubleshooting Guide for Journey Part III
Polling and Broadcast

Make a quick check on how many Treasures are in the World:
Go to the AgentCubes Windows menu and select "Show simulation Properties". This window will appear:

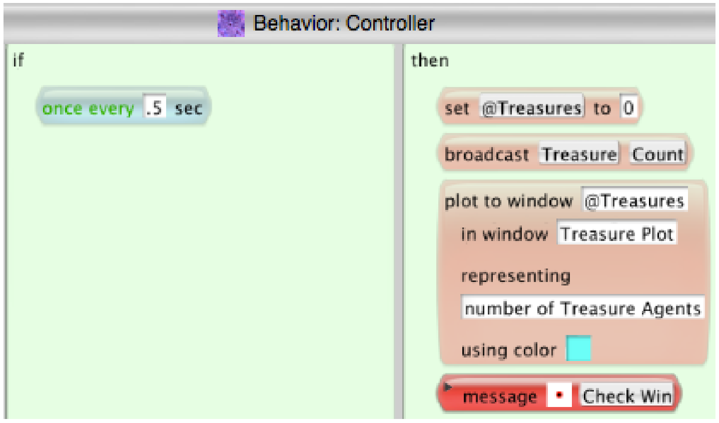


Simulation Properties

Property	Value
Treasures	5.0

The correct number of Treasures will not appear in this window until you have single-stepped (click on the black triangle next to the stop and go buttons) or briefly run the game. If your programming is correct, the value of Treasures will decrease by 1 each time your Traveler collects (erases) a Treasure. Stop the game and click on the word "Treasures" to check the current value of the Treasures simulation property. When the value of Treasures is equal to 0, you should win the game.

More detailed troubleshooting: To determine what is happening in your game, it is helpful to look at how the simulation property changes over time. Add the **plot to window** action to the rule in the Controller's **while running** method. Fill it out as it appears below:



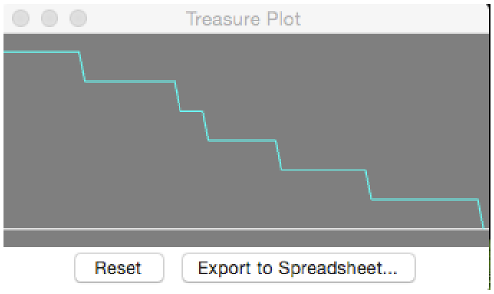
Behavior: Controller

```
if
    once every .5 sec
then
    set @Treasures to 0
    broadcast Treasure Count
    plot to window @Treasures
        in window Treasure Plot
        representing
        number of Treasure Agents
        using color 🟦
    message • Check Win
```

In the **plot to window** action, you must name the simulation property to be plotted (Treasures), name the window where it will appear (Treasure Plot), say what it represents (number of Treasure Agents) and pick the color of the line that will appear on the graph. **Note that you must put "@" before the Treasures in the plot to window action!**

DRAFT 3-D Journey (Continued)

Look under the AgentCubes Windows menu and select the Treasure Plot window. Move the Treasure Plot window somewhere where you can watch it while you run the game. In this window, you will see a graph that shows you what's happening 'behind the scenes' while you play the game.



This information will help you determine where a mistake may be. For example, if the number of treasures never goes above 0, there is a problem with the method Count or the broadcast. If the number of treasures goes to zero but the game doesn't end, there is a problem with the game ending rule in the Controller.

Student Handout 4:

Ice Arrows Challenge

Ice Arrows

Create arrows that freeze and unfreeze the Chasers

Before your start this challenge:

You must have a complete basic journey game with a Traveler who wins if s/he reaches the treasure and Chasers who chase the Traveler using a hill climbing search. The Traveler loses if a Chaser gets too close. The worksheet should have walls that the Traveler and Chasers can not cross.

Description of the Challenge:

- Your Traveler must shoot ice arrows up in all four directions (up, down, left and right).
- A Chaser hit by a moving ice arrow freezes and cannot move.
- A frozen Chaser hit by a moving ice arrow unfreezes and can move again.
- Ice arrows should not go through walls or stack up in piles.

Gamelet Design Activity:

In the description above, circle nouns to identify the agents and underline the verbs to identify actions associated with each agent. Mark adjectives to identify new shapes for an agent.

Create new agent: ice arrow

- Make it be an inflatable icon so you can draw your own picture.
- The picture for the ice arrow agent may face in any of the four directions.
- The point of the arrow should be a different color from the tail so that you can easily recognize which arrow you are seeing in the tiny pictures in the conditions.
- After you have drawn the first arrow shape, select the ice arrow agent and click on the +Shape button at the lower left corner of the AgentCubes window so you can draw an additional shape for the basic ice arrow.
- Draw 4 ice arrow shapes that so that the shapes face upwards, downwards, left and right.
- The ice arrow's shape stores its **Direction**. We can tell which way an ice arrow should move by checking its shape. The image saves the direction instead of an agent attribute.

Create a second shape for the Chaser: a frozen Chaser

- Select your chaser agent and click on the **+Shape** to create the frozen Chaser
- Make sure the frozen Chaser looks different enough that you can see it in the tiny pictures in the conditions.
- The Chaser's picture stores its **state**: frozen or unfrozen.

Traveler Design Challenges

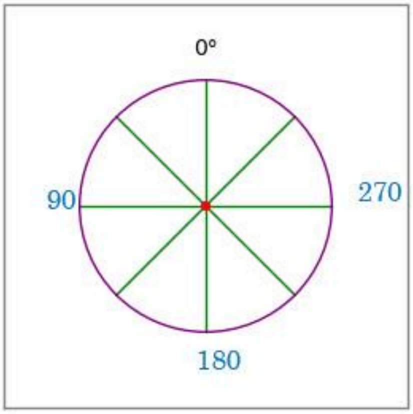
How do you know which ice arrow the Traveler should shoot?
The Traveler should shoot an ice arrow in the direction that s/he is facing.

How do you know which way the Traveler is facing?
The Traveler must have an Agent Attribute (or local variable) called Direction which keeps track of which way the Traveler is facing.
Initialize the Direction Agent Attribute in a when-creating-new-agent method.
When an arrow key is typed, set the Direction attribute and rotate the Traveler to face in that direction of the arrow key before the Traveler moves.

How can you avoid putting 4 rules for the 4 different ice arrows in the Traveler's while running method?
When the space bar is typed, make the Traveler send itself a message to do a method called Shoot Arrow that will shoot an arrow in the direction the Traveler is facing.

How are Directions named in AgentCubes?

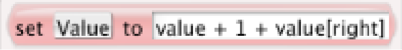
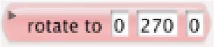
Directions must be named using degrees on a circle as in the picture below.



Up arrow key = move in 0 degrees direction
Left arrow key = move in 90 degrees direction
Down arrow key = move in 180 degrees direction
Right arrow key = move in 270 degrees direction

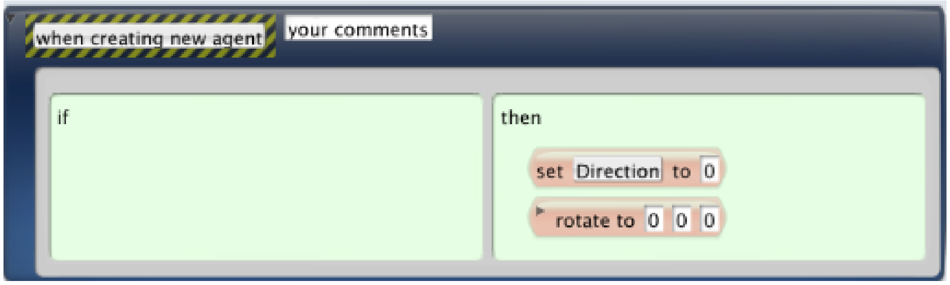
Traveler Rules

How to Create and Initialize the Traveler's Direction variable

1. Click on the +Method button
2. Click on the word on in the method's black and yellow tag and change it to "when-creating-new-method"
3. Drag in the set action  set Value to value + 1 + value[right] and make a new variable name, Direction.
4. Set Direction to 0.
5. Drag in the  rotate to 0 270 0 action and set all 3 numbers to 0.

6. Erase the Traveler and redraw the Traveler on the World, then save it.

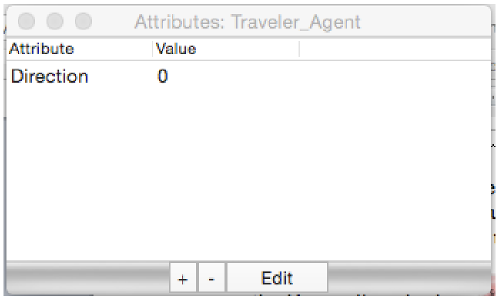
The Traveler's when-creating-new-agent method should look like this:



```
when creating new agent    your comments
if        then
            set Direction to 0
            rotate to 0 0 0
```

Test the value of the Traveler's Direction attribute

1. Choose **Show Agent Attributes** from the AgentCubes Window Menu
2. Click on the Traveler with the big Arrow tool.
3. You should see this:



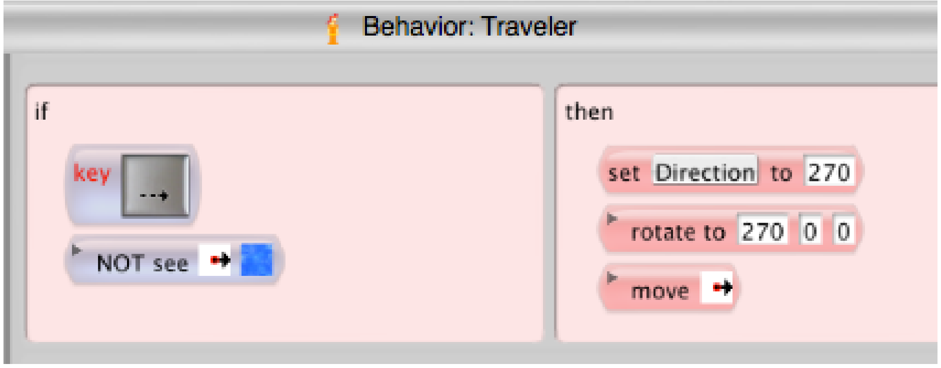
Attributes: Traveler_Agent

Attribute	Value
Direction	0

Update the Traveler's move rules

Add the **set** and **rotate to** actions to the Traveler's move rules so that the Traveler's Direction attribute changes as the player types the different arrow keys.

Here is a the move right rule. **Edit the other 3 move rules.**



```
Behavior: Traveler
if                    then
  key →                 set Direction to 270
  NOT see →             rotate to 270 0 0
                        move →
```

Note: the number for the direction goes in the <u>first</u> box of the rotate to action!

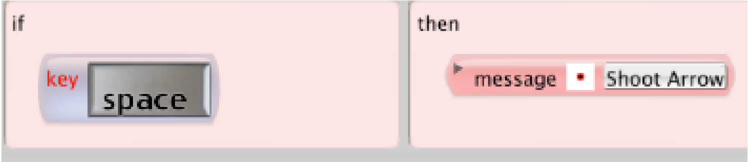
Test the Traveler's move rules to make sure the value of the Direction attribute matches the Traveler's movements

1. Choose **Show Agent Attributes** from the AgentCubes Window Menu
2. Click on the Traveler with the big Arrow tool.
3. Type the right arrow key. Does Direction change to 270 when the Traveler moves right? Does the Traveler rotate to face right?

4. Check the other arrow keys. **Remember move up is 0, move left is 90 and move down is 180.**

Making the Traveler Shoot Arrows

Here is the rule from the Traveler's while running method that generates ice arrows:

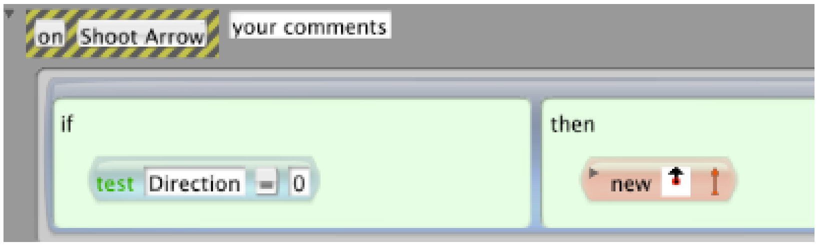


Where should this rule appear in the while running method box?

- Above or below the win rule?
- Above or below the move rules?

Remember: special cases and less common events appear above default behavior like moving!

Here is the first rule in the Traveler's Shoot Arrow method:



Direction refers to the Traveler's agent attribute Direction, which keeps track of which way the Traveler is facing. **Remember 0 is up, 90 is left, 180 is down and 270 is right.**

Make 3 more rules like this one for the remaining 3 directions.

Test your code:

- Does your Traveler generate ice arrows in all 4 directions?
  If not, make sure that your ice arrow pictures and the arrows in the new actions match the directions.

Ice Arrow Rules

Ice Arrow Design Challenges

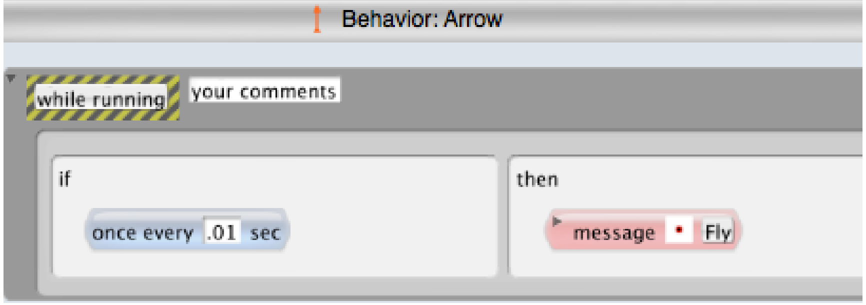
How do you know which direction the ice arrow should move?

Ice Arrows should move whichever direction they are pointing.

How can you avoid putting 4 rules for the 4 different ice arrows in the Ice Arrow's while running method?

Make the Ice Arrow send itself a message to do a method called Fly that will make an ice arrow move in the direction it points.

Here is the rule that belongs in the Ice Arrow's while running method:



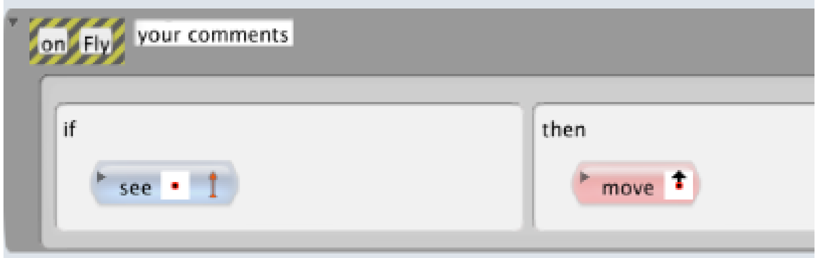
Behavior: Arrow

while running | your comments

if once every .01 sec

then message • Fly

Here is one of the 4 rules from the Ice Arrow's Fly method:



on Fly | your comments

if see • ↑

then move ↑

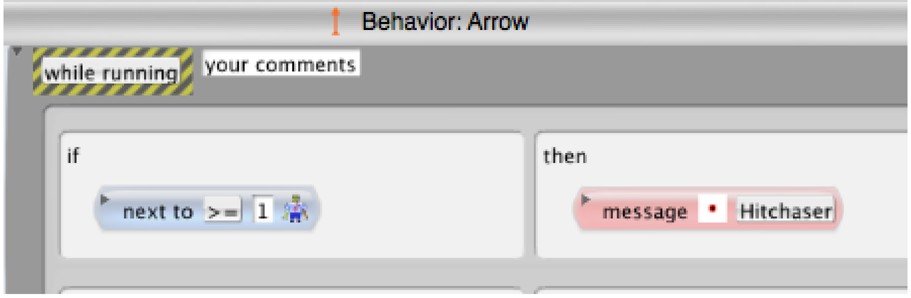
Add 3 more rules for the other 3 directions to the Fly method.

Test your code:

- Do your ice arrows move in all 4 directions?
  If not, make sure that your ice arrow picture and the move arrow match in each rule.

> **How do you make the point of the ice arrow hit the Chaser?**
> Only the point of the arrow can freeze or unfreeze the Chaser. It does not look convincing if the Chaser is hit by the side of an arrow. We need another method called HitChaser to see whether the arrow point will actually hit the Chaser.
> **What rules call HitChaser?**
> Use the **next to** condition to test whether the arrow is near a **frozen or unfrozen** Chaser, then use the HitChaser method to find out whether the Chaser is in front of the arrow point.

Here is the rule that calls HitChaser when the ice arrow is next to an ***unfrozen*** Chaser:



Behavior: Arrow

while running | your comments

if next to >= 1

then message • Hitchaser

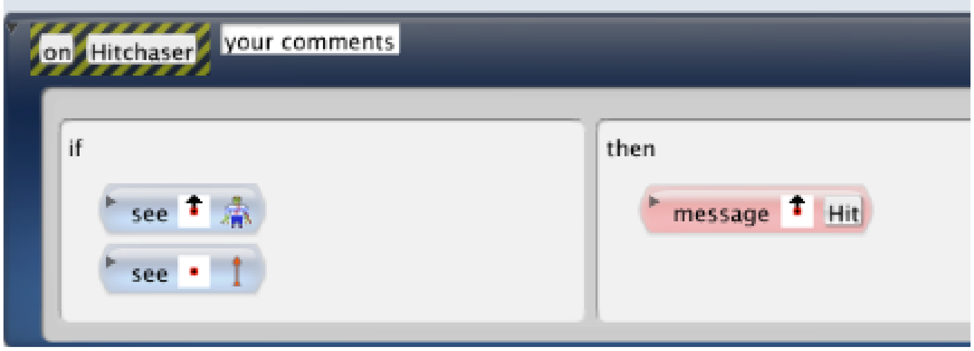
Does this rule belong above or below the rule that makes the ice arrow do it's Fly method? It's a special case and special cases should always be higher than other rules!

Make a second rule that will call HitChaser when the ice arrow is next to a *frozen* Chaser.

What do Ice Arrow's HitChaser rules do?

1. Test the shape of the arrow.
2. Test whether there is a Chaser on the grid square that the arrow points at.
3. Send a message to the Chaser that it has been Hit.

Here is a rule from the HitChaser method:

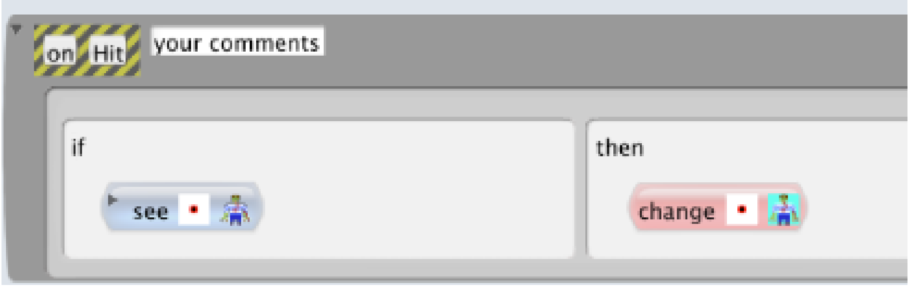


Make 3 more rules like this for the other 3 ice arrow shapes.
Then make the same 4 rules for the Frozen Chaser, so that it unfreezes if hit by an arrow.

Chaser Rules

How does the Chaser Freeze and Unfreeze?
The Hit method freezes an unfrozen Chaser and unfreezes a frozen Chaser.

Here is the rule from the Chaser's Hit method that **freezes** a Chaser:



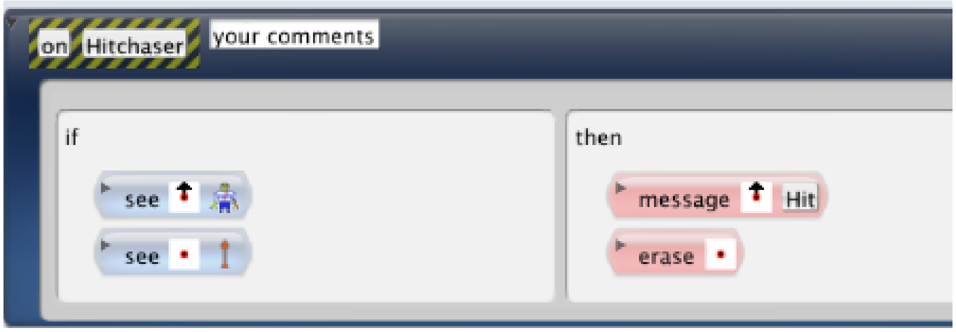
Note: The blue Chaser is frozen.

Make a second rule that will unfreeze a frozen Chaser.

Test your code:

- What happens when the Traveler shoots an ice arrow at the Chaser? You may need to temporarily delete the action in the Chaser's rules that makes it hill climb in search of the Traveler. It is much easier to test your ice arrows if the Chaser holds still!
- Does the ice arrow sit in front of the Traveler while the Traveler blinks back and forth between frozen and unfrozen shapes?
- The problem here is that the ice arrow keeps sending Hit messages to the Chaser so that the Chaser freezes and unfreezes over and over.
- How do you make the ice arrow stop sending messages to the Chaser?
- Erase it!
- Change the rules in the Ice Arrow's HitChaser method so that the ice arrow erases itself after it sends the Hit message to the Chaser.

Here is what the rules in HitChaser should look like now:

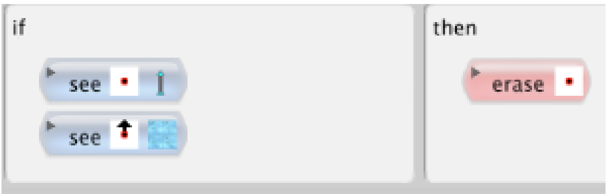


Test your code: does the Chaser stay frozen now until a 2nd ice arrow hits it?

One last step:

Do your ice arrows jump over walls?
Add rules to the Ice Arrow's Fly method so that ice arrows erase themselves if their points run into a wall.

Here is one of the rules:



Where does this rule go? Look →

Make 3 more rules like this for the other 3 ice arrow shapes.

Where should the rules about walls appear in the Ice Arrow's Fly method?

Special cases come before default behaviors!

1. Running into a wall happens less often so it is a special case and belongs above the regular fly rules.
2. The Ice Arrow's default behavior is to move forward so the call to the Fly method should be the last rule.

Test your program!

- Do your ice arrows stack up on the edges of your world?
- Put walls around the edges to absorb the ice arrows.
- Or create special ground agents that absorb ice arrows and put them around the edges of your world. If you choose this option, you will need 4 more rules in the Ice Arrow Fly method that look just like the wall rules but use the special ground agent instead of the wall. Put these rules just below the wall rules.
- Test that your Traveler is able to fire ice arrows in all 4 directions.
- The ice arrow should shoot in the Direction the Traveler is facing. If necessary, select the Traveler with the big arrow tool, then click on the Show Agent Attributes option in the AgentCubes Window menu to check which way the Traveler is facing and make sure that the ice arrow is fired the same direction. If there is a problem, go back through this tutorial and check that your rules exactly match the pictures.