

# ZOMBIE APOCALYPSE SIMULATION V12

## WHAT IS IT?

The agents in this model are HUMANS and ZOMBIES however, humans turn into zombies if they encounter a zombie and get bitten.

Zombies are reflex agents that always attack when confronted by a human. Humans are rational agents that make decisions based on their immediate environment and the actions they can take at that moment.

## HOW IT WORKS

There are two main agents in this model, Zombies and Humans. The environment spawns two objects randomly anywhere on the map to help assist the humans to defeat the zombies. Food to regain their health when necessary and weapons to have a fighting chance to defeat and kill the zombies. Food are denoted as a yellow fish icon and weapons are denoted as yellow 'x's.

In addition, the humans own the following abilities: Robustness to become strong, Speed variation to increase speed and Vision cone to be able to see the zombies and flee.

Furthermore, there are building/s that humans can pass through however, the zombies CANNOT pass through the buildings which makes the building a sort of SAFE ZONE for the humans.

If a zombie wins a fight and reduces the human's health to 0 then it bites the human, and the human turns into a zombie. Otherwise, the human may flee from the zombie or fight and kill the zombie using their weapon to avoid becoming infected.

## HOW TO USE IT

### Buttons

- SET-UP resets the simulation and to get ready
- GO (FOREVER) to run the simulation and watch in action

## Sliders

- number\_of\_humans to increase the amount of humans that are placed
- humans\_speed to increase the speed variation of the humans
- bwr to increase the range of the degree humans turn away from zombies
- vis\_rad to increase the radius of the vision cone
- vis\_ang to increase the angle of the vision cone
- number\_of\_zombie to increase the amount of zombies placed
- zombie\_speed to increase the speed variation of the zombies
- pwr to increase the range of the degree zombies turn to humans

## Switch

- show\_col\_rad to enable the radius around the human
- show\_vis\_cone to enable the vision cone function

## Monitors

- Zombies monitors the amount zombies are left
- Humans monitors the amount of humans left
- Food monitors the amount of food left
- Weapon monitors the amount of weapons left

## Credits

This script was created by Anthony Constant (AC). If you have any questions or suggestions, you can contact him at [anthonyconstant.co.uk/](http://anthonyconstant.co.uk/)

## License

This script is released under the MIT License. See the LICENSE file for more details.

## GitHub

Share Link:

## NETLOGO COPY & PASTED LOCAL SOURCE CODE

```
breed [ zombies zombie ]           ; creating a population of zombie who will move around aimlessly
breed [ humans human ]             ; creating a population of humans who will move around aimlessly but also seen the zombie
breed [ food fish ]                 ;creating a population of fish for food for the humans to eat to regain health!
breed [ weapon ammo ]               ; creating a population of weapons/ammo as weapons to defend the humans!

patches-own [solid ]                ;this creates a variable for the patches to establish if it should be percieved as solid

humans-own [ zombie_seen zombie_encounter
encountered
  health robustness speed_variation
  per_vis_rad per_vis_ang
  food_around_me closest_food
  have_weapon
  vis_rand
]
;this creates 2 variables which will be used to count the total zombies seen and zombies
;this creates 3 variables for health, durability and speed
;this creates variables for personlised vision cones
;this creates 2 variables to save the location of food
;this creates a variable to store the amount of weapon held
;this creates a variable to store a stable vision cone random value

zombies-own [
  human_around_zombie
  closest_human
]
;this creates a variable for the zombie to detect a human in it's radius
;this creates a variable to detect the closest human

food-own [ amount ]                 ;this creates a variable for the food to establish amount of the resource

globals [rad
daytime starting_color current_color
color_adjust color_range
  timer_reset ]
;this creates a global variable called rad
;this creates 3 global variables relating to creating day and night within our model
;this creates a global variable called for resetting the timer

to setup
  clear-all
  reset-ticks
  set rad 5
; this creates a function called setup
; this clears the world of any previous activities
; this resets the ticks counter
; this sets the global variable rad to 3
```

```

set timer_reset 1000           ;this sets the global variable reset_timer to 1000
set daytime true               ;this sets the global variable daytime to true
set starting_color 95          ;this sets the global variable starting_color to 85 which is blue
set current_color starting_color ;this sets the global variable current_color to starting_color
set color_range 5              ;this sets the global variable color_range to 5.
set color_adjust ( color_range / ( timer_reset + 10 )) ;this sets the global variable color_adjust to a range based on the variable
above

```

```

create-zombies number_of_zombie [           ; this creates the number of zombie that your global variable states
  setxy random-xcor random-ycor             ; this sets the starting position of the zombie to a random location in the world
  set color gray                             ; this sets the color of the zombie to gray
  set size 10                               ; this sets the size of the zombie to 10
  set shape "person"                        ; this sets the shape of the zombie to a person
]

```

```

create-humans number_of_humans [; this creates the number of humans that your global variable states
  setxy random-xcor random-ycor             ; this sets the starting position of the humans to a random location in the world
  set color red                             ; this sets the color of the humans to blue
  set size 10                               ; this sets the size of the humans to 10
  set shape "person"                       ; this sets the shape of the humans to a human

```

```

set health 30 + random 10           ;sets the health of the human by adding 50 + a random allocation up to 50
adjust_vision_cone                  ;set up the vision cone
set robustness random 10            ;sets the robustness variable to a random value up to 10
set speed_variation random 10       ;higher the number the faster they will go
;set heading 0                     ; demonstrate it has impact
;pen-down                          ; this puts the pen down to see where the human moves (history of the human)
set vis_rand random 5

```

```

  ifelse show-health?               ;show-health? switch
  [ set label health ]              ;show the health stat for humans
  [ set label "" ]                  ;set string label

```

```

]

```

```

create-weapon 10 [                 ;this creates X number of new weapons for the humans to store and use against the zombies
  make_weapon                       ;this calls the make_weapon function
]

```

[illegible]

```

end

to grow_food ;this creates a function called grow_food
  setxy random-xcor random-ycor ;this sets the position of the food to a random location in the world
  set color yellow ;this sets the color of the food to yellow
  set size 10 ;this sets the size of the food to 10
  set shape "fish" ;this sets the shape of the food to a fish
  set amount random 10 ;this sets the amount of food per plant to a random value up to ' '
end

to go ; this creates a function called go
  make_zombie_move ; this calls the make_zombie_move function
  reset_patch_color ; this calls the reset_patch_color function
  make_humans_move ; this calls the make_humans_move function
  draw_building ; calls the draw building function
  tick ; this adds 1 to the tick counter
  grow_more_food ; this calls the grow_more_food function
  if not any? humans [ stop ] ; exits if there are no more humans
  if not any? zombies [ stop ] ;exits if there are no more zombies

end

to make_zombie_move ; this creates a function called make_zombie_move
  ask zombies[ ; this asks all of the zombie in the population to do what is in the brackets
    set color gray ; this sets the color of each person to gray

    let can_see_human human_functions 30 ;set can_see_human radius to 30
    ifelse ( can_see_human = true ) [ ;if can_see_human is true then...
      set heading (towards closest_human ) ;set zombie heading towards the closest human
    ]
    [right ( random pwr - ( pwr / 2))] ; this turns the person right relative to its current heading by a random degree number
    using the range set within pwr NOTE: if negative it will turn left

    detect_wall ;this calls the detect_wall function
    forward zombie_speed ; this sets the speed at which the zombie move
  ]
end

```

```

to-report human_functions [sensitivity]
  expects a value for sensitivity
  set human_around_zombie other ( humans in-radius sensitivity )
  human within the sensitivity radius
  set closest_human min-one-of human_around_zombie [ distance myself ]
  human source
  let can_see_human [false]

  if (closest_human != nobody ) [
    set can_see_human true
  ]
  report can_see_human
called
end

to reset_patch_color
  ifelse daytime = true [
    set current_color current_color - color_adjust
  ][
    set current_color current_color + color_adjust
  ]
  ask patches [
    if solid = false [
      set pcolor current_color
    ]
  ]
end

to make_humans_move
  ask humans [
    ifelse health > 0 [
      show_visualisations
      set color red
      let have_seen_zombie human_function
    ]
  ]
  it with the return

```

;this creates a reporting function called human\_functions and  
 ;this sets the human\_around\_zombie variable to the ID's of the  
 ;this sets the closest\_human variable to the ID of the closest  
 ;set can\_see\_human to false  
 ;if closest\_human is equal to nobody then...  
 ;set can\_see\_human to true  
 ;return value of can\_see\_human to location where function was  
 ;this creates a function called reset\_patch\_color  
 ;if global variable daytime is true...  
 ;adjust global variable current\_color using color\_adjust variable  
 ;otherwise...  
 ;adjust global variable current\_color using color\_adjust variable  
 ; this asks all of the patches in the population to do what is in  
 ; this sets the color of each patch to current\_color  
 ;this is defining a function called make\_humans\_move  
 ;this asks all of the humans in the population to do what is in  
 ;if health is greater than 0 ( still alive)...  
 ;call the show\_visualtions function  
 ;this sets the color of each human to red  
 ;this creates a local variable called have\_seen\_zombie the fills  
 it with the return

<pre>         let can_smell_food food_function 30 it with the return value         pickup_weapon          ifelse ( have_seen_zombie = true ) [             right 180         ][             ifelse ( can_smell_food = true ) [                 set heading ( towards closest_food )             ][                 right (random bwr - (bwr / 2)) random degree number             ]              forward humans_speed + ( speed_variation * 0.01 )         ][             set color gray             convert         ]      ]  end  to show_visualisations     if show_col_rad = true [ if the switch is set to true         ask patches in-radius rad [ the collision radius             if solid = false [                 set pcolor orange             ]         ]     ]      if show_vis_cone = true [ the switch is set to true         ask patches in-cone per_vis_rad per_vis_ang [ </pre>	<pre> ;this creates a local variable called can_smell_food then fills ;this calls the pickup_weapon function  ;if local variable have_seen_zombie is true... ;set the heading of human to 180 (turn around to avoid zombie!)  ;if local variable can_smell_food is true... ;set heading towards closest food source  ;this turns the human right relative to its current heading by a  ;moves human forward by the humans_speed variable  ;set color to gray to indicate dead human ;this kills the human off  ; this creates a new function called show_visualisation ; this will switch on the visualisation of the collision radius  ; this sets up a radius around the zombie to display the size of  ; this sets the patch color to orange  ; this will switch on the visualisation of the vision cone if  ; this sets up a vision cone to display the size of the cone by </pre>
---	--



changing the patch color	
if solid = false [	
set pcolor red	; this sets the patch color to red
]	;+++++++ closing if statement
]	
end	
to-report food_function [sensitivity]	;this creates a reporting function called food_function abd
expects a value	
set food_around_me other ( food in-radius sensitivity )	
set closest_food min-one-of food_around_me [distance myself]	
let can_smell_food [false]	;this creates a local variable valled can_smell_food and sets it
to false	
let eating_food [false]	
if health < 100 [	;if health is less than 100 then...
ask food in-radius rad [	;this sets up a radius around the food to the value of the
global variable rad	
ifelse amount > 0 [	;if amount is greater than 0
set eating_food true	;set the local variable called eating_food to true indicating
the human is eating	
set amount amount - 5	;reduces 5 from the amount variable in the food
set color color - .25	;reduce the color intensity of the food by .25
][	
die	;there is no food left so kill the agent
]	
]	
if eating_food = true [	;if eating_food is true then...
set health health + 5	;add 5 health to the human
]	
if (closest_food != nobody) [	;if closest_food is not empty (the human can smell food in range)
then...	
set can_smell_food true	;set can_smell_food to true
]	
report can_smell_food	;return value of can_smell_food to location where function is

```
being called
end
```

```
to-report human_function
```

```
    let seen [false] ; this creates a local variable called seen
    let hit [false] ; this creates a local variable called hit
    let zombie_hit 0 ;this creates a local variable calls upon zombie_hit and sets
it to 0

    ask zombies in-cone per_vis_rad per_vis_ang [ ; this sets up a vision cone with the parameters from vis_rad
vis_ang to detects zombie
        set color green ; this sets the color of the person detected within the vision
code of the human to green
        set seen true ; this sets the local variable called seen to true indicating
that a person has been seen
    ]

    ask zombies in-radius rad [ ; this sets up a radius around the human for collision
detection with zombie using rad
        set hit true ; this sets the local variable called hit to true indicating
that a person has collided with
        set zombie_hit who ;this sets the local variable called person_hit to the
individual who
    ]

    ifelse seen = true [ ; if then else statement based on the local variable seen, if
seen = true then...
        set zombie_seen zombie_seen + 1 ; add 1 to the zombie_seen count
        set color green ; set color of human to white
        ;right 180 ; set heading of the human to 180 (turn around to avoid!)
    ][ ; if seen = false...

        ;right (random bwr - (bwr / 2)) ; this turns the human right relative to its current heading by a
random degree number using the range set within bwr NOTE: if negative it will turn left
    ]

    if hit = true [ ; if statement based on the local variable hit, if seen = true
```

```

then...
  ifelse have_weapon > 0 [
    ask zombie zombie_hit [die]
    set have_weapon have_weapon - 1
  ][
    set zombie_hit zombie_hit + 1
    set color green
    set health health - robustness
    adjust_vision_cone
  ]
]
report seen ;+++++
end

to pickup_weapon
  let pickup [false]
false
  ask weapon in-radius rad [
global variable rad which we are using for collision detection with the weapons to pick it up
    set pickup true
    die
  ]
  if pickup = true [
    set have_weapon have_weapon + 1
  ]
end

to adjust_vision_cone
check if health drop below 0 ( error checking )
  if (((vis_rad + vis_rand)*(health * 0.01))) - ((starting_color - current_color) * 2) > 0) [
what they can see is being reduced
    set per_vis_rad (((vis_rad + vis_rand)*(health * 0.01)) - ((starting_color - current_color) * 2))
vision radius to factor in some randomness and health ( less health = less vision )
  ]
  if ((vis_ang + vis_rand)*(health * 0.01)) > 0 [
is greater than 0 then...
    set per_vis_ang ((vis_ang + vis_rand)*(health * 0.01))
vision angle to factor in some randomness and health ( less health = less vision)
  ]

```

;if have\_weapon is greater than 0 then...  
 ;kills of the zombie hit  
 ; remove 1 from the have\_weapon of the human  
  
 ; add 1 to the zombie\_encounter count  
 ;if hit by a zombie set human colour to green  
 ;+++++  
 ;+++++

; this creates a function called pickup\_weapon  
 ;this creates a loval variable called pickup and sets it to  
  
 ;this sets up a radius around the human to the value of the  
 ;this sets thew local variable pickup to true

;if pick is true then...  
 ;add 1 to the have\_weapon count on the human

;if statement is to  
 ;if not as healthy  
  
 ;set the pesonal  
  
 ;if the calculation  
  
 ;set the personal

```

]
end

to grow_more_food ;this creates a new
function called grow_more_food
  if ticks > timer_reset [ ;+++++
ticks is greater than 100 then... ;if the current numer of
  ask patch random-xcor random-ycor [ ;ask the patch to do the
following... ;sprout (create new) food
    sprout-food 1 [grow_food]
(1 in this instance) then call grow_food function to set the parametres
  ]
  ifelse daytime = true [ ;if global variable
daytime is true then... ;set global variable
    set daytime false
daytime to false ;otherwise...
  ][ ;set global variable
    set daytime true
daytime to true
  ]
  reset-ticks ; this resets the tick
counter back to default
]
end

```





