

Rules of the Game: A model of *Anthopleura*
elegantissima clonal formation

Robert T. Lindsay

PO Box 14680, Stanford, CA 94309

6/14/06

Advisors: Chuck Baxter, Susan Shillinglaw, William Gilly

Permission is granted to Stanford University to use the citation and abstract of this paper.

Abstract:

The anemone *Anthopleura elegantissima* inhabits the mid to high intertidal on the Pacific coast of North America. It reproduces asexually by longitudinal fission, eventually forming large colonies of genetically identical clonemates. There is evidence of social organization within *A. elegantissima* clones; Ayre and Grosberg (2005) identified five “castes” that polyps belong to: “scouts,” “warriors,” “reproductives,” “free-edge,” and a fifth caste of small interior polyps mixed in with the large “reproductives.” Ayre and Grosberg did not know the function of this latter group, positing that they could be the result of “unusual episodes of asexual reproduction,” or former scouts retreating into the interior to heal. I created a program in MATLAB to model the formation of these clones, based on simple rules for growth and division. I found that based on these rules, small interior polyps developed consistently alongside the larger “reproductives.” This finding gives evidence to the hypothesis that variability in size is a completely natural result of asexual reproduction. Future models could show that variability of the number of acrorhagi (fighting tentacles) and gonads is also a natural result of the interaction of individual polyps with the environment, and complex intracolony signaling mechanisms are unlikely and unnecessary to explain diversity within a clone. Consequently, the assumption that every polyp plays a well-defined role in the colony may be misguided.

Introduction:

The sea anemone *Anthopleura elegantissima* inhabits the mid to high intertidal at Hopkins Marine Station and elsewhere along the Pacific coast. It reproduces both sexually, with gonads, and asexually via longitudinal fission. By asexual reproduction it forms large colonies, or “clones,” of genetically identical individual polyps, slowly attempting to take over the surrounding inhabitable area. Occasionally two clones will expand until they are next to each other, forming a border, or “no man’s land,” and will subsequently fight over the space. There is evidence of large variation in size, number of acrorhagi (fighting tentacles), and number of gonads of a polyp, depending on its location in an aggregation. Within each fighting colony, Ayre and Grosberg (2005) identified five “castes”:

- Scouts: small polyps that dart into the “no man’s land”
- Warriors: small polyps with large numbers of acrorhagi and small numbers of gonads, that line the border between an opposing colony and engage in aggressive searching behavior
- Reproductives: large polyps located in the interior of the colony, with large numbers of gonads and small numbers of acrorhagi, that appear to play the most prominent role in sexual reproduction
- Free-edge: small polyps about the same size as warriors, but with fewer acrorhagi. These polyps are located on the edge of the colony, but do not oppose another colony

- A fifth caste, comprised of small polyps interspersed with the “reproductives” in the interior of the colony

Ayre and Grosberg identified polyps located in the interior and weighing less than 2 grams as candidates to be included in the fifth caste. They note that 36% of polyps in the interior, defined as 7-10 polyp-widths from the interclonal border, are this small (compared to a total average of about 4.5 grams for all polyps in this region). However, their exact role in the colony is unclear to Ayre and Grosberg: “neither the origin, nor function, of this category of polyp are obvious...the presence of these polyps may reflect unusual episodes of asexual reproduction...Alternatively, these small polyps may be recycled scouts that have retreated deep into the clone following fighting”¹ (emphasis my own.)

Materials and Methods:

I created a computer program in MATLAB® designed to model the formation and expansion of *Anthopleura elegantissima* colonies. The environment is a 64x64 two-dimensional grid, and each cell is either black (nothing living there) or a shade of green (inhabited by a polyp). Since anemones are fairly two-dimensional, modeling them on a plane does not sacrifice much spatial information. The model begins with a single 1x1 polyp in the middle, and then applies rules in a succession of rounds. There are two basic rules, growth and division. In every round, each polyp has a chance to grow, and a chance to divide. It is important to note that these probabilities correspond to an attempt

¹ Ayre and Grosberg, pg. 107

at growth or division, and the outcome is uncertain. For example, if a polyp is bounded on every side by its neighbors, it has no room to grow or divide, so any attempts at growth or division would fail. The probability of division is based upon a laboratory study by Sebens (1980). This study investigated the total number of polyps in a colony of a particular size, and also the number of polyps in a colony of a particular size that were dividing. To obtain a probability vector for division, I sampled the size chart at 7 intervals of size, and divided the number of polyps undergoing asexual reproduction with the total number of polyps (see Figs. 1 & 2.)

In this model, the ovular and circular forms of anemones are approximated by square cells on a grid (see Fig. 3). The basal diameter is interpreted to be the length plus the width of a polyp, divided by two (for example, a 3x2 polyp would be interpreted to have a basal diameter of 2.5). Polyps divide by longitudinal fission, so a 3x2 polyp would divide 66-33 into a 2x1 polyp and a 2x2 polyp. A 2x4 polyp would divide 50-50 into two 2x2 polyps. This is similar to the natural world, where a study found that the larger of two daughter clones is between 51 and 75% of the original weight (Francis, 1976). In the first 10 runs, the probability of growth is uniform across all polyps, i.e., it does not depend on an individual's size. In two additional runs, the probability of growth is weighted such that the expected net increase in size was the same for all polyps. Growth is always perpendicular to the longer side, so a 3x2 polyp that is growing would turn into a 3x3 polyp. In the case that the polyp is square, the angle of the growth is

random. The actual direction (i.e. whether the polyp grows or divides to the left or right) is also random. In the case of asymmetrical division, the larger daughter clone is always oriented in the same direction to ensure that the model does not skew sizes to be larger in the interior or exterior of the colony.

Results:

The model was first run on 10 different growth probabilities, varying from 10% to 100% chance of a growth attempt each round, in a 64x64 field. Each run was allowed to stabilize, i.e. to expand to all four corners of the grid. Small polyps were identified as those having a size 1x1 or 1x2. To compare this to the small polyps (<2g) in Ayre and Grosberg, one has to convert area into volume, an equation complicated by the hollowness of anemones in general. In any case, 1x2 and 1x1 polyps are significantly smaller than the largest (3x4) polyps, a 1x2 polyp being 1/6 of the total area. In all 10 runs of the model, these interior small polyps proliferated. This data is presented in Fig. 4.

The mean percentage of interior small polyps is 27%, with a maximum of 35% and a minimum of 18%. In Fig. 5, the polyps marked with red dots are those of the smaller size. There were two additional runs with probabilities of division weighted to account for the gross gain in size dependant on the size of the growing polyp. For example, a 1x1 polyp growing will add another 1 cell to itself, while a 3x3 polyp growing will add 3 additional cells to itself. The probability weights were calculated to have an expected net increase that was uniform across all polyps. In one run where the expected

gain was 1 cell per polyp per round, the percentage of interior small polyps was 31%. In a run where the expected gain was .5 cells per polyp per round, the percentage of interior small polyps was 30%.

Discussion:

One should not read too much into the actual number of the percentage of small polyps; although the mean of 27% in the model vs. the 36% found by Ayre and Grosberg is similar, it is in different units (area of rectangular polyps vs. wet weight of cylindrical polyps). However, the take-home message is that there is a lot of variability in the size of interior polyps, and there are substantial numbers of polyps on the lower end of the size spectrum. This indicates that the origin of many, if not all, of Ayre and Grosberg's "fifth caste" could be the effects of completely *usual* asexual reproduction in a constrained area. In this model, with very simple rules of division and growth, smaller polyps are produced alongside the larger ones. The other hypothesis of the origin of these polyps, that they are "recycled scouts," seems unlikely given that they witnessed this occurring only once. One way to find out would be to take long-term video of an aggregation and track the growth and division of polyps over time; however, this would take many years.

Given that the fifth caste described by Ayre and Grosberg could be a natural result of asexual reproduction in general, whether this caste has a special "origin" or "function" in the clone is a misleading question. Many of the specializations in *A. elegantissima* colonies could emerge from simple rules of growth, division, and acrorhagial stimulation that apply to all polyps, no matter their location within the colony. In the same paper,

Ayre and Grosberg found that exposing reproductive polyps to acrorhagial stimulation by a non-clonemate could induce the polyp to grow more acrorhagi. Based on this simple premise, that attacked polyps are likely to become attackers themselves, this model could be developed to demonstrate how “warriors” and “reproductives” could also emerge from simple, universal rules. Ayre and Grosberg comment that “...in the case of *A. elegantissima* aggregations, communication between clonemates appears to be critical for maintaining or developing the preparedness of clones for the defence or acquisition of habitable space.”² The simple rules used in this model are a much simpler mechanism for the variability found within the colony than intracolony communication. One reason that intracolony communication is unlikely is that its closest relative, *Anthopleura sola*, is solitary and asexual, therefore lacking the function of communication for altruistic purposes.

Conclusions:

Diversity within an aggregation of *Anthopleura elegantissima* can be produced with a model using very simple, universal rules. We get many interesting properties, such as small polyps interspersed with large ones in the interior, “for free.” This is a simpler explanation for the emergence of so-called castes than intracolony signaling, and, according to Occam’s razor, a more likely one.

On a deeper level, trying to find the “origin” or “purpose” of every polyp in the clone is a misguided exercise. Some insights into this problem are provided by John Steinbeck’s book *The Log from the Sea of Cortez*. In chapter 14, there is an extended

² *Ibid*, pg. 107.

diatribe, originally written by Steinbeck's friend Ed Ricketts, on the subject of poisonous teleological thinking. An example he gives is the question, "why are some matches larger than others?"³ Steinbeck/Ricketts argue:

The differences will group into plus-minus variations from a hypothetical mean to which not one single example will be found exactly to conform. Now the ridiculousness of the question becomes apparent. There is no *particular* reason. It's just so. There may be in the situation some factor or factors more important than the others: owing to the universality of variation (even in those factors which "cause" variation), there surely *will* be, some even predominantly so. But the question as put is seen to be beside the point. The good answer is: "it's just in the nature of the beast." (pg. 114)

Similarly, in the case of *Anthopleura elegantissima* clonal formation, much of the social organization that has been described in the literature does not need to be attributed to a specific purpose that benefits the aggregation as a whole, but can be understood to be the result of individual polyps interacting with their surrounding environment in simple, predetermined ways.

Acknowledgements:

I would like to thank William Gilly, Susan Shillinglaw, Chuck Baxter, Fiorenza Micheli, and Charles Hanifin for their help with the project and their leadership of the Holistic Biology course. I also want to thank Bruce R. Land for letting me use sections of his artificial life code in MATLAB® in my model.

³ Steinbeck, pg. 114.

Literature Cited:

Ayre, D. J. and R. Grosberg. 2005. Behind anemone lines: factors affecting division of labour in the social cnidarian *Anthopleura elegantissima*. *Animal Behavior* 70:97-110.

Francis, L. 1976. Social organization within clones of the sea anemone *Anthopleura elegantissima*. *Biological Bulletin* 150:361-376.

Sebens, K.P. 1980. The regulation of asexual reproduction and indeterminate body size in the sea anemone *Anthopleura elegantissima* (Brandt). *Biological Bulletin* 158:370-382.

Steinbeck, John. *The Log from the Sea of Cortez*. New York: Penguin Books, 1995.

Appendix A: Figures

Fig. 1. Number of total polyps and dividing polyps in a population of Anthopleura elegantissima as a function of size. From Sebens, 1980.

Fig. 2. Probability of division as a function of polyp size used in the model.

Fig. 3. Sample run of the model

Fig. 4. Percentage of interior small polyps as a function of growth probability (uniform)

Fig. 5. Small interior polyps marked with a red dot in a run of the model

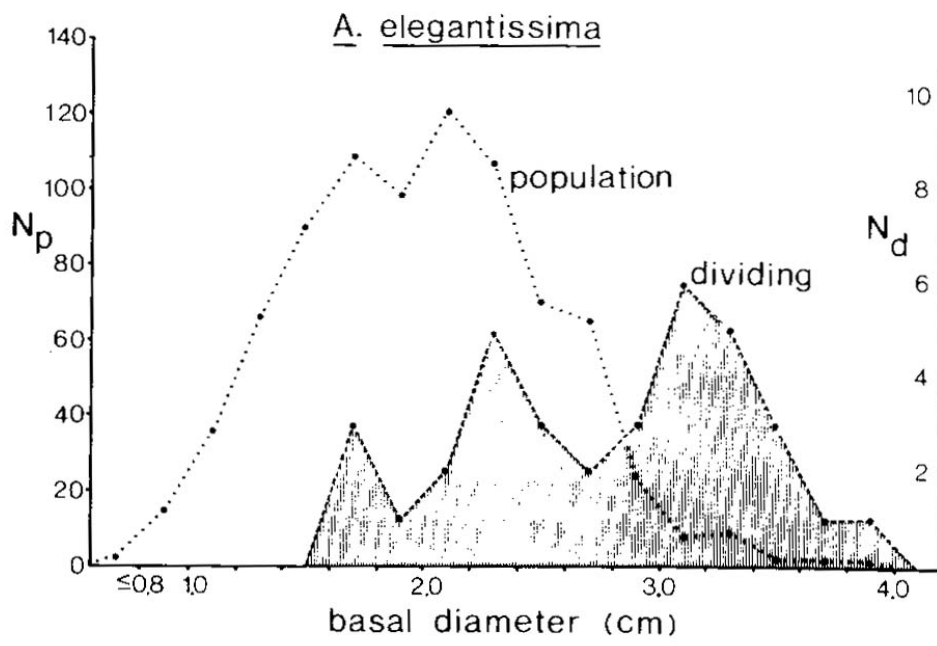


Fig. 1

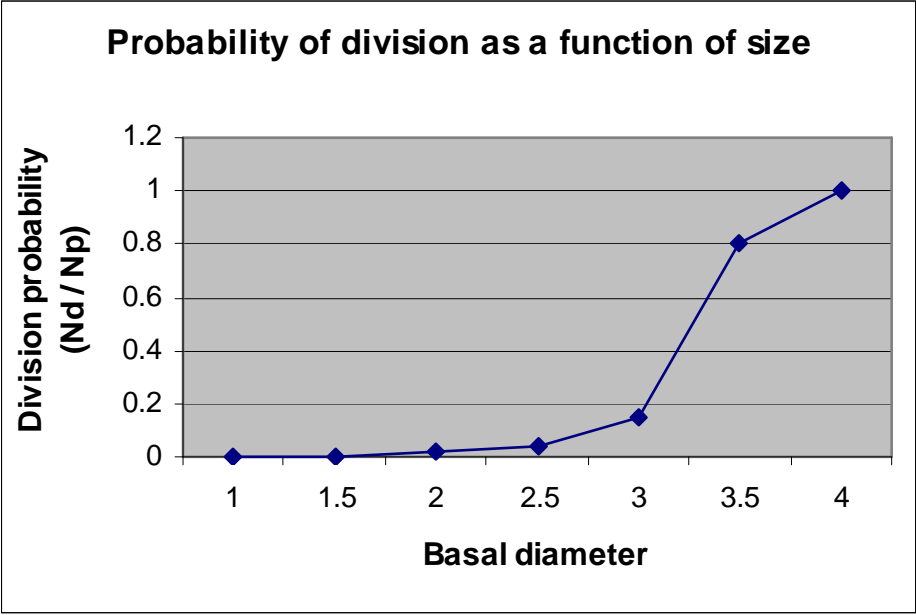


Fig. 2



Fig. 3

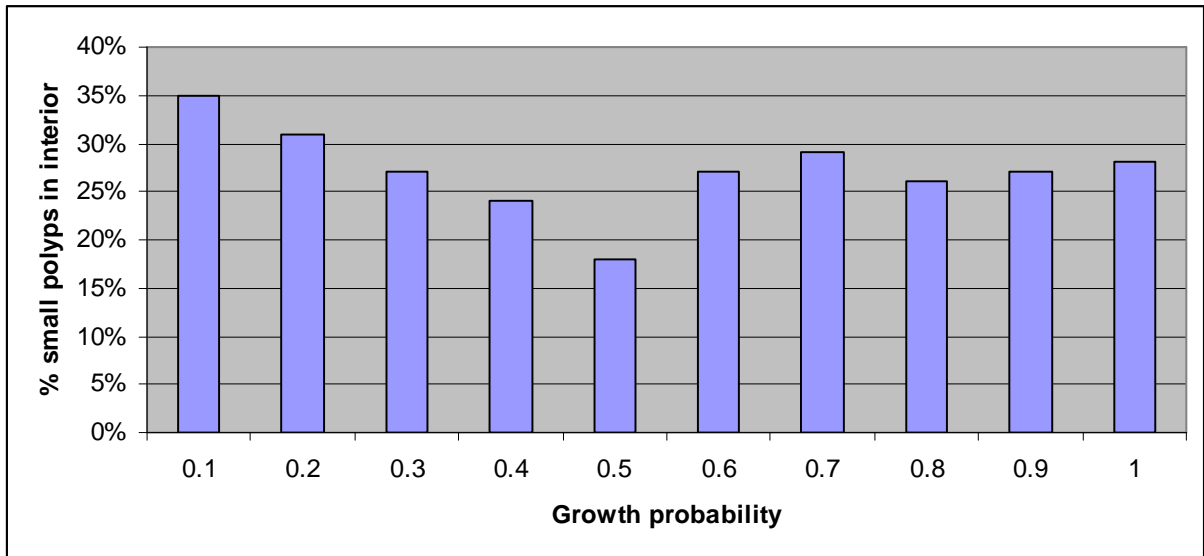


Fig. 4

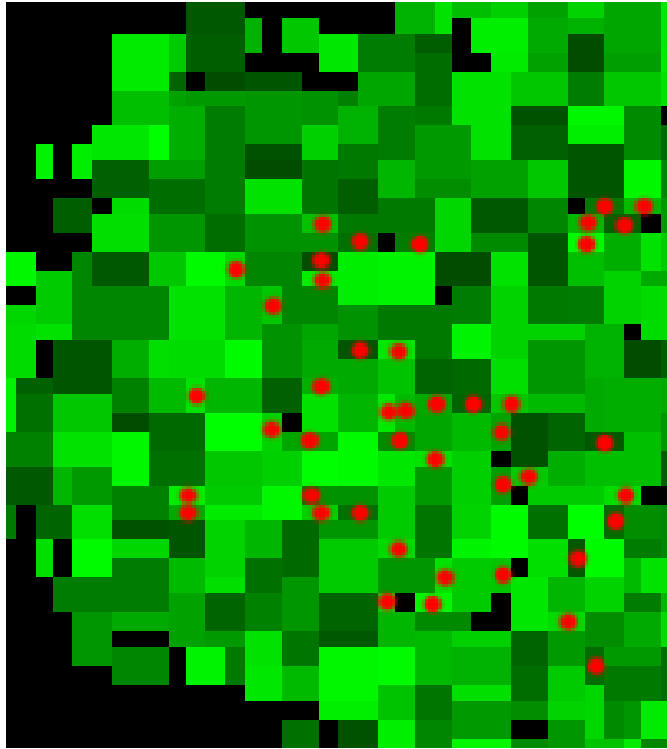


Fig. 5

Appendix B: Source Code

Note: significant portions of the GUI and runtime engine of this model were adapted from Bruce Land's MATLAB model of artificial life, used with permission. His website can be found at <http://www.nbb.cornell.edu/neurobio/land/>.

```
% Anthopleura elegantissima clonal formation model

clf
clear all

%=====
%build the GUI
%define the plot button
plotbutton=icontrol('style','pushbutton',...
    'string','Run', ...
    'fontsize',12, ...
    'position',[100,400,50,20], ...
    'callback', 'run=1;');

%define the stop button
erasebutton=icontrol('style','pushbutton',...
    'string','Stop', ...
    'fontsize',12, ...
    'position',[200,400,50,20], ...
    'callback','freeze=1;');

%define the Quit button
quitbutton=icontrol('style','pushbutton',...
    'string','Quit', ...
    'fontsize',12, ...
    'position',[300,400,50,20], ...
    'callback','stop=1;close;');

quitbutton=icontrol('style','pushbutton',...
    'string','Stats', ...
    'fontsize',12, ...
    'position',[400,400,50,20], ...
    'callback','freeze=1;stats=1;');

number = uicontrol('style','text', ...
    'string','1', ...
    'fontsize',12, ...
    'position',[20,400,50,20]);

%=====

% SETTINGS
% size of board
n=64;

% how much delay (in seconds) for each round
delay = 0;

% division probability vector based upon sampling data from Sebens, 1980
probs = [0,0,0,2/50,3/70,3,20,4/5,1];

% master growth probability variable
master_gp = .9;
```

```

% growth probability vector, uniform across all sizes
%g_probs =
[master_gp, master_gp, master_gp, master_gp, master_gp, master_gp, master_gp, master_gp, master_g
p];
%g_probs = [1, .5, .5, .33, .33, .25, .25, .20, .20];
g_probs = [.5, .25, .25, .16, .16, .125, .125, .1, .1];
% probability of "recovery time" following division (1 = no delay)
delay_prob = 1;

% number of buckets for statistical analysis (alpha)
num_buckets = 7;

for a = 1:num_buckets
    buc_chart(a) = 0;
    buc_count(a) = 0;
end

max_dist = 0;

%initialize the arrays
z = zeros(n,n);

% initialize the field
for a = 1:n
    for b = 1:n
        cells(a,b).alive = 0;
        cells(a,b).is_master = 0;
        cells(a,b).master_loc = [];
        cells(a,b).num_slaves = 0;
        cells(a,b).upper_left = [];
        cells(a,b).lower_right = [];
        cells(a,b).color = 0;
        cells(a,b).blue_color = 0;
        cells(a,b).delay = 0;
    end
    size_chart(a) = 0;
    chart_count(a) = 0;
end

%init boundaries (alpha)

% for a = 1:n
%     for b = 1:n
%         %if(((a + b) < n | (a + b) > 4*n/3) & (abs(a-b) > 4))
%         %if(a+b < 4*n/5 | b < n/4 | (a > 3*n/4 & (b + 3*n/4) > a))
%         if(b > n/2)
%             cells(a,b).alive = 1;
%             cells(a,b).is_master = 0;
%             cells(a,b).master_loc = [];
%             cells(a,b).num_slaves = 0;
%             cells(a,b).upper_left = [];
%             cells(a,b).lower_right = [];
%             cells(a,b).color = 0;
%             cells(a,b).blue_color = 1;
%         end
%     end
% end

% set middle cell to alive at the start
x = n/2;
y = n/2;
start = [x,y];
cells(x,y).alive = 1;
cells(x,y).upper_left = [x,y];
cells(x,y).lower_right = [x,y];
cells(x,y).is_master = 1;

% set random color, but make sure it's visible
cells(x,y).color = rand;

```

```

while(cells(x,y).color < .3)
    cells(x,y).color = rand;
end

%build an image and display it

for Xco = 1:n
    for Yco = 1:n
        grid(Xco,Yco) = cells(Xco,Yco).alive * cells(Xco,Yco).color;
        boundary_grid(Xco,Yco) = cells(Xco,Yco).blue_color;
    end
end

imh = image(cat(3,z,grid,boundary_grid));
set(imh, 'erasemode', 'none')
axis equal
axis tight

%index definition for cell update
x = 2:n-1;
y = 2:n-1;

%Main event loop
stop= 0; %wait for a quit button push
run = 0; %wait for a draw
freeze = 0; %wait for a freeze
stats = 0;

while (stop==0)
    pause(delay);
    if (run==1)

        % main loop
        for Xco = 1:n
            for Yco = 1:n
                % only do something if cell is a master
                if(cells(Xco,Yco).is_master == 1)

                    % diagonal vector, which is the size of the polyp
                    diag = cells(Xco,Yco).lower_right - [Xco Yco] + [1,1];

                    % pick a random sign, to expand either green or down
                    if(rand < .5)
                        sign = -1;
                    else
                        sign = 1;
                    end

                    avg_size = (diag(1) + diag(2));

                    % get probabilities for growth and division, based upon size of
                    % the polyp
                    div_prob = probs(avg_size);
                    growth_prob = g_probs(avg_size);

                    % if there is delay in the polyp, chance to eliminate it
                    if(cells(Xco,Yco).delay >= 1)
                        if(rand < delay_prob)
                            cells(Xco,Yco).delay = cells(Xco,Yco).delay - 1;
                        end
                        % don't do anything this round
                        growth_vector = [0 0];
                        % chance for division first
                        elseif(rand < div_prob)
                            if(diag(1) > diag(2) | (diag(1) == diag(2) & rand < .5))
                                growth_vector = [diag(1)*sign 0];
                            else
                                growth_vector = [0 diag(2)*sign];
                            end
                            growth = 0;
                        % chance for growth

```

```

elseif(rand < growth_prob)
    if(diag(1) > diag(2) | (diag(1) == diag(2) & rand < .5))
        growth_vector = [0 diag(2)*sign];
    else
        growth_vector = [diag(1)*sign 0];
    end
    growth = 1;
% not growing or dividing, then do nothing
else
    growth_vector = [0 0];
end

% dummy variable for checking neighboring cells for OOB indices and
% living cells
c = 1;

% vertical polyp expanding upwards
if(growth_vector(1) < 0)
    for num = 0:(diag(2)-1)
        cell_to_be_checked_x = cells(Xco,Yco).upper_left(1) - 1;
        cell_to_be_checked_y = cells(Xco,Yco).upper_left(2) + num;

        % check OOB and if something is living
        % in that spot

        if(cell_to_be_checked_x < 1 | cell_to_be_checked_x > n | ...
            cell_to_be_checked_y < 1 | cell_to_be_checked_y > n | ...
            cells(cell_to_be_checked_x,cell_to_be_checked_y).alive == 1)
            c = 0;
            break;
        end
    end
    if(c == 0)
        % if blocked, skip 'er
        continue ;
    else
        % good to go
        % change checked cells into living
        % create polyp gap by setting midline cells
        % into dead

        newcolor = rand;
        for num = 0:(diag(2)-1)
            change_x = Xco - 1;
            change_y = Yco + num;
            cells(change_x,change_y).alive = 1;
            cells(change_x,change_y).color = cells(Xco,Yco).color;
            if(growth == 0)
                erase_x = cells(Xco,Yco).lower_right(1) -
floor(abs(growth_vector(1)/2));
                cells(erase_x,change_y).alive = 0;
            end
        end

        % change lower_right and upper_lefts
        temp_ul = cells(Xco,Yco).upper_left;
        temp_lr = cells(Xco,Yco).lower_right;
        cells(Xco,Yco).is_master = 0;
        cells(temp_ul(1) - 1, temp_ul(2)).upper_left = temp_ul + [-1 0];
        cells(temp_ul(1) - 1, temp_ul(2)).is_master = 1;
        % dividing
        if(growth == 0)
            cells(temp_ul(1) - 1, temp_ul(2)).delay = 1;
            cells(temp_ul(1) - 1, temp_ul(2)).lower_right = ...
                [temp_lr(1) - floor(abs(growth_vector(1)/2)) - 1, temp_lr(2)];
            cells(temp_lr(1) - floor(abs(growth_vector(1)/2)) + 1
,temp_ul(2)).is_master = 1;
            cells(temp_lr(1) - floor(abs(growth_vector(1)/2)) + 1 ,temp_ul(2)).delay
= 1;

```

```

        cells(temp_lr(1) - floor(abs(growth_vector(1)/2)) + 1
,temp_ul(2)).upper_left = ...
        [temp_lr(1) - floor(abs(growth_vector(1)/2)) + 1, temp_ul(2)];
        cells(temp_lr(1) - floor(abs(growth_vector(1)/2)) + 1
,temp_ul(2)).lower_right = temp_lr;
        % update colors on new polyp
        newcolor = rand;
        while(newcolor < .3)
            newcolor = rand;
        end

        ul = [temp_lr(1) - floor(abs(growth_vector(1)/2)) + 1 ,temp_ul(2)];
        lr = temp_lr;

        for k = ul(1):lr(1)
            for kk = ul(2):lr(2)
                cells(k,kk).color = newcolor;
            end
        end

        % not dividing, just growing
    else
        cells(temp_ul(1) - 1, temp_ul(2)).lower_right = temp_lr;
    end
    continue;
end
end

%vertical polyp expanding downwards
if (growth_vector(1) > 0)
    for num = 0:(diag(2)-1)
        cell_to_be_checked_x = cells(Xco,Yco).lower_right(1) + 1;
        cell_to_be_checked_y = cells(Xco,Yco).upper_left(2) + num;

        % check OOB and if something is living
        % in that spot

        if(cell_to_be_checked_x < 1 | cell_to_be_checked_x > n | ...
            cell_to_be_checked_y < 1 | cell_to_be_checked_y > n | ...
            cells(cell_to_be_checked_x,cell_to_be_checked_y).alive == 1)
            c = 0;
            break;
        end
    end
    if(c == 0)
        % if blocked, skip 'er
        continue;
    else
        % good to go
        % change checked cells into living
        % create polyp gap by setting midline cells
        % into dead

        %growth = 1;

        newcolor = rand;
        for num = 0:(diag(2)-1)
            change_x = cells(Xco,Yco).lower_right(1) + 1;
            change_y = cells(Xco,Yco).upper_left(2) + num;
            cells(change_x,change_y).alive = 1;
            cells(change_x,change_y).color = cells(Xco,Yco).color;

            % delete midline if cell is dividing
            if(growth == 0)
                erase_x = Xco + floor(growth_vector(1)/2);
                cells(erase_x,change_y).alive = 0;
            end
        end
    end

    % change lower_right and upper_lefts

```

```

temp_ul = cells(Xco,Yco).upper_left;
temp_lr = cells(Xco,Yco).lower_right;
new_ul_x = Xco + floor(growth_vector(1)/2) + 1;

% if cell is dividing
if(growth == 0)
    cells(Xco,Yco).lower_right = [temp_ul(1) - 1 + ...
        floor(growth_vector(1)/2), temp_lr(2)];
    cells(Xco,Yco).delay = 1;
    cells(new_ul_x, temp_ul(2)).is_master = 1;
    cells(new_ul_x, temp_ul(2)).delay = 1;
    cells(new_ul_x, temp_ul(2)).upper_left = ...
        temp_ul + floor(growth_vector/2) + [1 0];
    cells(new_ul_x, temp_ul(2)).lower_right = temp_lr + [1 0];

    % update colors on new polyp
    newcolor = rand;
    while(newcolor < .3)
        newcolor = rand;
    end

    ul = temp_ul + floor(growth_vector/2) + [1 0];
    lr = temp_lr + [1 0];

    for k = ul(1):lr(1)
        for kk = ul(2):lr(2)
            cells(k,kk).color = newcolor;
        end
    end
else
    cells(Xco,Yco).lower_right = temp_lr + [1 0];
end
continue;
end
end

% horizontal polyp expanding leftwards
if (growth_vector(2) < 0)
for num = 0:(diag(1)-1)
    cell_to_be_checked_x = cells(Xco,Yco).upper_left(1) + num;
    cell_to_be_checked_y = cells(Xco,Yco).upper_left(2) - 1;

    % check OOB and if something is living
    % in that spot

    if(cell_to_be_checked_x < 1 | cell_to_be_checked_x > n | ...
        cell_to_be_checked_y < 1 | cell_to_be_checked_y > n | ...
        cells(cell_to_be_checked_x,cell_to_be_checked_y).alive == 1)
        c = 0;
        break;
    end
end
if(c == 0)
    continue;
else
    % good to go
    % change checked cells into living
    % create polyp gap by setting midline cells
    % into dead

    for num = 0:(diag(1)-1)
        change_x = cells(Xco,Yco).upper_left(1) + num;
        change_y = cells(Xco,Yco).upper_left(2) - 1;
        cells(change_x,change_y).alive = 1;
        cells(change_x,change_y).color = cells(Xco,Yco).color;

        % delete midline if cell is dividing
        if(growth == 0)
            erase_y = cells(Xco,Yco).lower_right(2) + floor(growth_vector(2)/2);
            cells(change_x,erase_y).alive = 0;

```

```

        end
    end

    % change lower_right and upper_lefts
    temp_ul = [Xco Yco];
    temp_lr = cells(Xco,Yco).lower_right;

    cells(Xco,Yco).is_master = 0;
    cells(temp_ul(1), temp_ul(2) - 1).upper_left = temp_ul + [0 -1];
    cells(temp_ul(1), temp_ul(2) - 1).is_master = 1;

    % if dividing
    if(growth == 0)
        cells(temp_ul(1), temp_ul(2) - 1).delay = 1;
        cells(temp_ul(1), temp_ul(2) - 1).lower_right = temp_lr +
floor(growth_vector/2) + [0 -1];
        cells(temp_ul(1), temp_lr(2) + floor(growth_vector(2)/2) + 1).is_master =
1;
        cells(temp_ul(1), temp_lr(2) + floor(growth_vector(2)/2) + 1).delay = 1;
        cells(temp_ul(1), temp_lr(2) + floor(growth_vector(2)/2) + 1).upper_left
= ...
        [temp_ul(1), temp_lr(2) + floor(growth_vector(2)/2) + 1];
        cells(temp_ul(1), temp_lr(2) + floor(growth_vector(2)/2) + 1).lower_right
= temp_lr;

    % update colors on new polyp
    newcolor = rand;
    while(newcolor < .3)
        newcolor = rand;
    end

    ul = [temp_ul(1), temp_lr(2) + floor(growth_vector(2)/2) + 1];
    lr = temp_lr;

    for k = ul(1):lr(1)
        for kk = ul(2):lr(2)
            cells(k,kk).color = newcolor;
        end
    end
else
    % just growing, not dividing
    cells(temp_ul(1), temp_ul(2) - 1).lower_right = temp_lr;
end
continue;
end
end

% horizontal polyp expanding rightwards
if (growth_vector(2) > 0)
    for num = 0:(diag(1)-1)

        cell_to_be_checked_x = cells(Xco,Yco).upper_left(1) + num;
        cell_to_be_checked_y = cells(Xco,Yco).lower_right(2) + 1;

        % check OOB and if something is living
        % in that spot

        if(cell_to_be_checked_x < 1 | cell_to_be_checked_x > n | ...
            cell_to_be_checked_y < 1 | cell_to_be_checked_y > n | ...
            cells(cell_to_be_checked_x,cell_to_be_checked_y).alive == 1)
            c = 0;
            break;
        end
    end
    if(c == 0)
        % if blocked, skip 'er
        continue;
    else
        % good to go
        % change checked cells into living

```

```

newcolor = rand;
for num = 0:(diag(1)-1)

    change_x = cells(Xco,Yco).upper_left(1) + num;
    change_y = cells(Xco,Yco).lower_right(2) + 1;
    cells(change_x,change_y).alive = 1;
    cells(change_x,change_y).color = cells(Xco,Yco).color;

    % if dividing
    % create polyp gap by setting midline cells
    % into dead
    if(growth == 0)

        erase_y = Yco + floor(growth_vector(2)/2);
        cells(change_x,erase_y).alive = 0;
    end
end

% change lower_right and upper_lefts
temp_ul = cells(Xco,Yco).upper_left;
temp_lr = cells(Xco,Yco).lower_right;
new_ul_y = Yco + floor(growth_vector(2)/2) + 1;

% if dividing
if(growth == 0)
    cells(Xco,Yco).lower_right = [temp_lr(1), Yco + floor(growth_vector(2)/2)
- 1];

    cells(Xco,Yco).delay = 1;
    cells(Xco, new_ul_y).is_master = 1;
    cells(Xco, new_ul_y).delay = 1;
    cells(Xco, new_ul_y).upper_left = [Xco, new_ul_y];
    cells(Xco, new_ul_y).lower_right = temp_lr + [0 1];

    % update colors on new polyp
    newcolor = rand;
    while(newcolor < .3)
        newcolor = rand;
    end

    ul = [Xco, new_ul_y];
    lr = temp_lr + [0 1];

    for k = ul(1):lr(1)
        for kk = ul(2):lr(2)
            cells(k,kk).color = newcolor;
        end
    end
    % just growing, not dividing
else
    cells(Xco,Yco).lower_right = temp_lr + [0 1];
end
continue;
end
continue;
else
    continue;
end
end
end
% done, go to next point
end

% update grid with new colors
for Xco = 1:n
    for Yco = 1:n
        grid(Xco,Yco) = cells(Xco,Yco).alive * cells(Xco,Yco).color;
    end
end

%draw the new image

```

```

set(imh, 'cdata', cat(3,z,grid,boundary_grid))
%update the step number display
stepnumber = 1 + str2num(get(number, 'string'));
set(number, 'string', num2str(stepnumber))
end

if (freeze==1)
if(stats==1)
% show statistics (alpha)
small = 0;
total = 0;
total_size = 0;
small_size = 0;
for Yco = 1:n
for Xco = 1:n
if(cells(Xco,Yco).is_master)
size = cells(Xco,Yco).lower_right(1) - Xco + cells(Xco,Yco).lower_right(2) -
Yco + 1;

midpoint = [(Xco + cells(Xco,Yco).lower_right(1))/2, (Yco +
cells(Xco,Yco).lower_right(2))/2];
rawdist = floor(abs(midpoint(1) - start(1)) + abs(midpoint(2) - start(2)));
if(size <=2 & rawdist < n/2)
small = small + 1;
total = total + 1;
small_size = small_size + size;
total_size = total_size + size;
elseif(rawdist < n/2)
total = total + 1;
total_size = total_size + size;
end
if(rawdist > max_dist)
max_dist = rawdist;
end
if(rawdist > 0)
size_chart(rawdist) = size_chart(rawdist) + size;
chart_count(rawdist) = chart_count(rawdist) + 1;
end
end
end
end

% more statistics (optional)
disp('percent of small polyyps');
small/total
disp('average size of small polyyps');
small_size/small
disp('average size in interior');
total_size/total

for bucket_num = 1:(num_buckets-1)
for dist = 1:n
if(dist > ((bucket_num) * max_dist)/num_buckets & dist <=
((bucket_num+1)*max_dist)/num_buckets)
buc_chart(bucket_num) = buc_chart(bucket_num) + size_chart(dist);
buc_count(bucket_num) = buc_count(bucket_num) + chart_count(dist);
end
end
end

for k = 1:num_buckets
if(buc_count(k) > 0)
%average out the data
buc_chart(k) = buc_chart(k)/buc_count(k);
end
end
plot(buc_chart);
end
run = 0;
freeze = 0;
end

```

```
drawnow %need this in the loop for controls to work  
end
```