P4

edgardjfc

March 2022

1 Introduction

For this assignment we used Voronoi diagrams to represent a 2d plane of crystal grains. The objective was to create two cracks within this plane of grains, and make it so the cracks are biased towards traveling within the grain boundaries, then making several repetitions of this and observing the probability of both cracks converging with one another. Finally, making a statistical graph that shows the effects of the seed density on this probability.

2 Code

```
def propagate(vor):
    # initial parameters
    prob, difficulty = 0.9, 0.8
    (x, y) = initialise()

    # results
    crackPoints = []

    while True:
        # crack this point
        crackPoints.append((x,y))

        # determine which neighbours belong to the same cell (interior), and which belong to
        boundary, interior = [], []
        for v in neighbours:
            (dx, dy) = v
            vx, vy = x + dx, y + dy
            if vx >= 0 and vx < n and vy >= 0 and vy < n: # inside the image
               if not (vx,vy) in crackPoints: # no crack
                    if vor[vx, vy] == vor[x, y]: # same cell
                        interior.append(v)
                    else:
                        boundary.append(v)
```

```
        # select a point to spread to
        selected = None
        if len(boundary) > 0:
            selected = choice(boundary)
            prob = 1
        elif len(interior) > 0:
            selected = choice(interior)
            prob *= difficulty

        # if a point is selected, then spread
        if selected is not None:
            (dx, dy) = selected
            x, y = x + dx, y + dy
        else:
            break # stops spreading

    return crackPoints
```

This function is similar to the original function developed by our teacher Dr. E. Shaeffer [1], however, there are important changes made to it. One big change is that the points of the crack are no longer drawn in the image as they are created, but instead they are saved as coordinates on lists, to be drawn on the image later.

```
def simulate(seedDensity):
    # generate a list of random seeds
    seeds = []
    for s in range(seedDensity):
        while True:
            x, y = randint(0, n - 1), randint(0, n - 1)
            if (x, y) not in seeds:
                seeds.append((x, y))
                break

    # generate the initial image
    cells = [cell(i, seeds) for i in range(n * n)]
    voronoi = Image.new('RGB', (n, n))
    vor = voronoi.load()
    c = sns.color_palette("Set3", seedDensity).as_hex()
    for i in range(n * n):
        vor[i % n, i // n] = ImageColor.getrgb(c[cells.pop(0)])

    # propagate two cracks across the voronoi diagram
    crack1 = propagate(vor)
    crack2 = propagate(vor)
```

This function's job is to call the propagate function two different times, in order to create two lists of points that will be later used to color the image with the cracks and also use the list to see if there is an overlap within the two cracks.

3 results

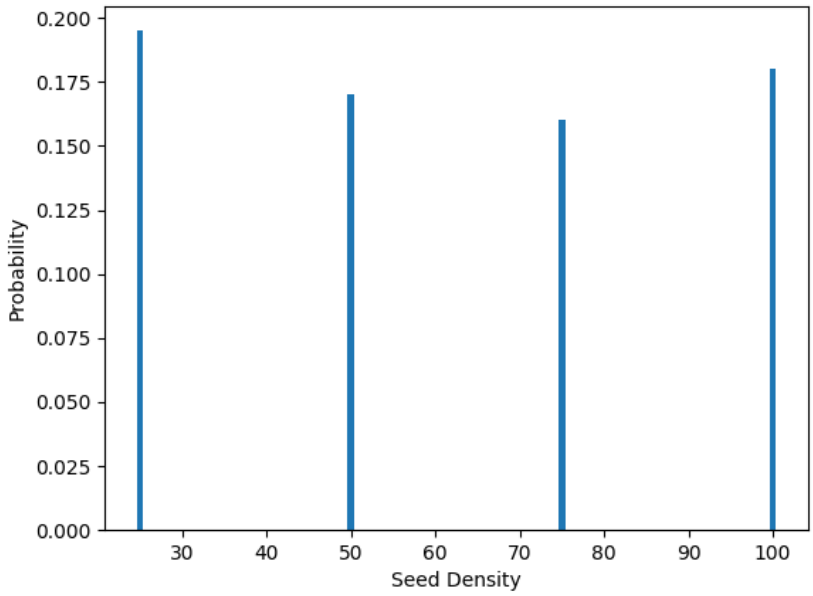

Figure 1: Plotted data

4 Interpretation

In our data plotted in fig. 1 we can observe how the probability for fractures to meet is decreasing as the seed density increases. This is coherent with the way metals work, where the mechanical properties of a metal improve the smaller the grains are, due to there being more grain boundaries to disperse the cracks and deformations.

5 Conclusion

We observed how the cracks behaved similarly to how they behave in actuality, this is because of how the code was written, given the cracks a bias towards

Figure 2: Cracks do not touch

Figure 3: Cracks touch

spreading in between the grain boundaries, the same way it happens in real life materials.

References

[1] E. Schaeffer. *GitHub,fracture.py*. URL: `https://github.com/satuelisa/Simulation/blob/master/VoronoiDiagrams/fracture.py`.