CGT: Graph Analysis

Tirthankar Adhikari

Task: Given several graphs obtained by plotting the N and P positions (Black and White dots) of different Combinatorial games, it is observed that the area under the graph is divided into several distinct regions based on the Texture formed by the dots and the task is to separate out these regions and comment on the slopes of the boundaries formed between them.

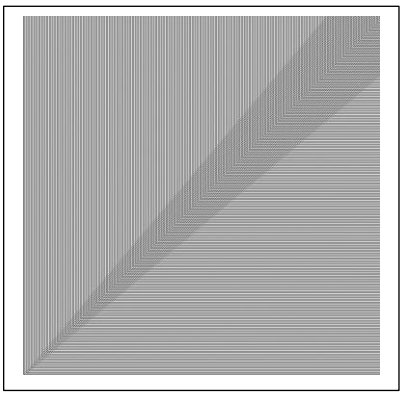
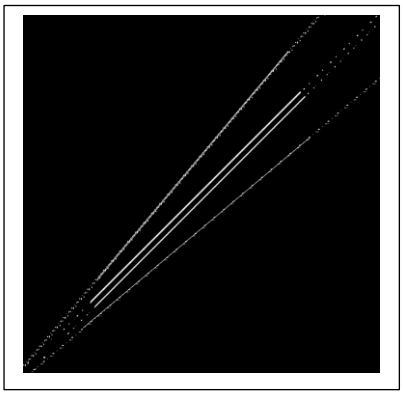
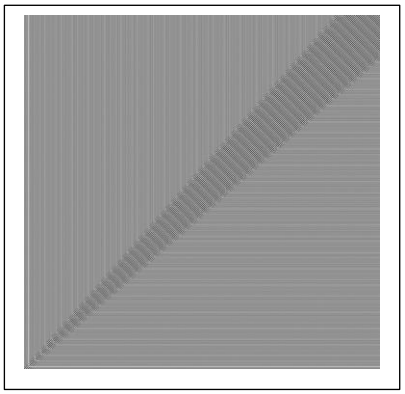
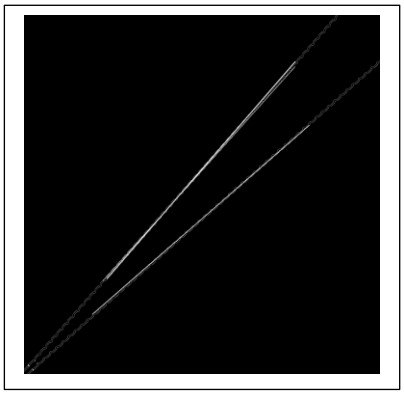
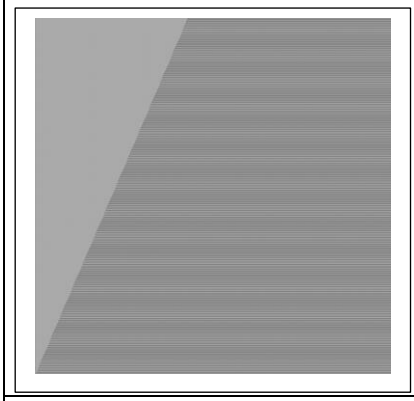
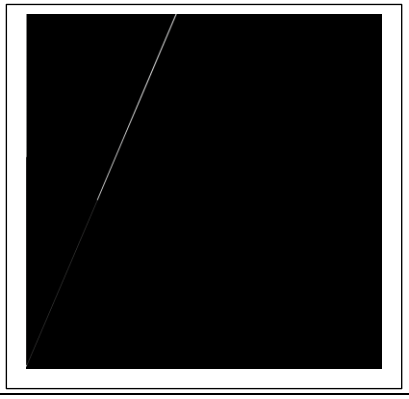
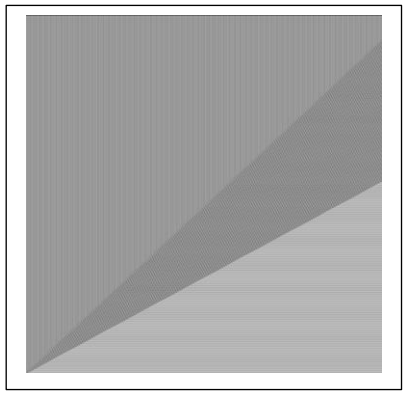
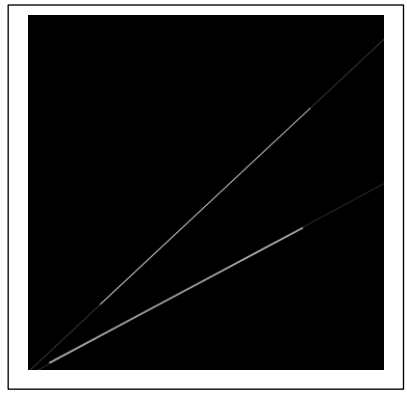
Approach: Texture analysis refers to the characterization of regions in an image by their texture content. Texture analysis attempts to quantify intuitive qualities described by terms such as rough, smooth, silky, or bumpy as a function of the spatial variation in pixel intensities. In this sense, the roughness or bumpiness refers to variations in the intensity values, or gray levels. Texture analysis is used in various applications, including remote sensing, automated inspection, and medical image processing. Texture analysis can be used to find the texture boundaries, called **texture segmentation**.

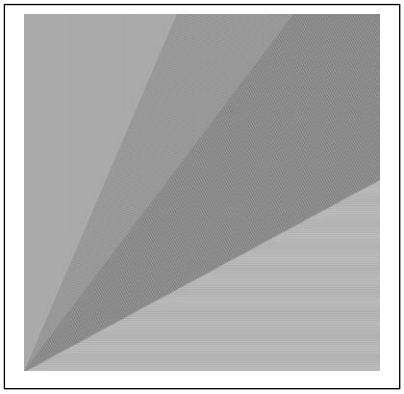
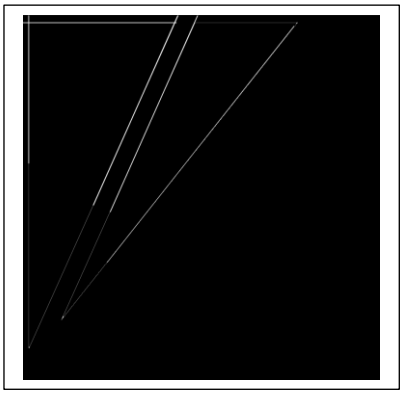
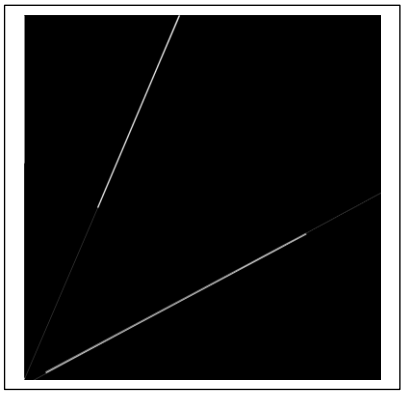
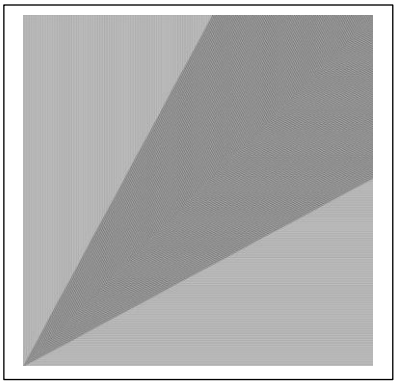
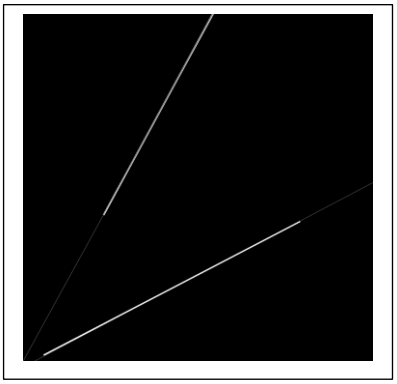
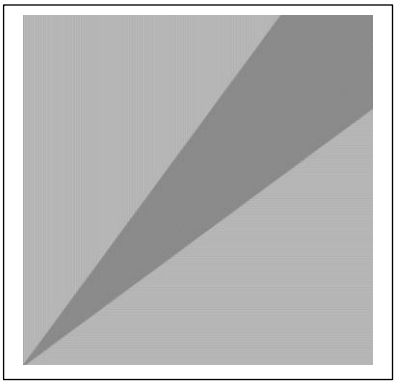
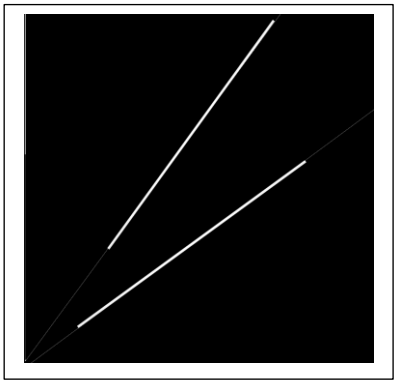
For this task we will be using 2 filters and 1 transformation, namely:

1) **Variance filter:** Highlights edges in the image by replacing each pixel with the neighborhood variance.
2) **Entropy filter:** Entropy is a statistical measure of randomness that can be used to characterize the texture of the input image. This filter can segment portions of an image by their complexity.
3) **Hough transform:** A popular technique for detecting straight lines and curves on gray-scale images. It maps image data from image space to a parameter space, where curve detection becomes peak detection problem.

The code that has been written is based on the above 2 filters and the Hough transform has its final result dependent on 3 parameters:

1) **n :** The no. of times the variance filter is applied
2) **l :** The window size of the variance filter
3) **k :** The no. of times entropy filter is applied

Input image	Parameters	Output image	Slopes
	n = 1 l = 35 k = 2		[1.0, 1.1912908 242612752, 0.8387728 459530026, 1.1929961 089494163, 1.0, 1.0, 1.0, 1.0, 1.0]
	n = 1 l = 5 k = 2		[0.8688741 721854305 1.1492764 661081494, 0.8688741 721854305, 1.1501524 390243902, 1.1105302 464525766]
	n = 1 l = 5 k = 1		[2.3529411 764705883, 2.35294117 64705883]
	n = 1 l = 5 k = 1		[0.9323308 270676691, 0.9329685 3625171, 0.5317100 792751982, 0.5320113 314447592, 0.5317100 792751982]

	n = 3 l = 30 k = 1 n = 1 l = 5 k = 1	 	[1.3264119601328903, 2.35595390524968, 2.35595390524968, 2.35595390524968, 2.3529411764705883] [2.3529411764705883, 2.3529411764705883, 2.3529411764705883, 0.5317100792751982, 0.5320113314447592, 0.5317100792751982]
	n = 1 l = 5 k = 1		[1.8796592119275826, 0.5317100792751982, 1.8796592119275826, 1.8796592119275826, 0.5317100792751982, 0.5320113314447592, 0.5317100792751982]
	n = 1 l = 5 k = 1		[1.3758503401360545, 1.3770212765957446, 1.3770212765957446, 1.3767006802721109, 1.3770212765957446, 0.7268232385661311, 0.7262051915945612, 0.7262051915945612, 0.7262051915945612, 0.7268232385661311, 0.7262051915945612, 1.3758503401360545, 0.7268232385661311]

Future work: