109590004 呂育瑋 Machine Vision HW#3

這次作業實作以 Quad-tree 資料結構做影像處理，對圖像做灰階化與二值化後以不同的 Quad-tree Layer 輸出影像。

步驟如下：

對圖像做灰階化後，依題目要求的門檻值對影像作二值化，計算圖像最大 Layer 數為 log2(圖像邊長)，由 Layer 1 到最大 Layer 數依序輸出圖像。

```
vector<uchar> getBinaryImageValueList(const Point2i& pointBegin, const Point2i& pointEnd)
```

依照選取的區塊取得二值化圖像中的值，並回傳這些值的數列，用作區塊單色、非單色判斷依據。

```
uchar getMergeColor(const vector<uchar> pixels)
```

由 `getBinaryImageValueList()` 回傳的數列，將數列中的值加總後的平均值，判斷應該回傳 0, 128, 255 其中之一，如果加總平均值為 0 或 255，回傳加總平均值(即 0 或者 255)；否則回傳 128(表示區塊內非單色組成)。

```
void updateQuadTreeImage(const Point2i& pointBegin, const Point2i& pointEnd, const uchar& color)
```

依照指定的區塊為圖像填上 color 這個顏色，color 由 `getMergeColor()` 決定。

```
void updateQuadTreeRecursively(int treeHeight, const Point2i pointBegin, const Point2i pointEnd)
```

以遞迴函式建構 Quad-tree，treeHeight 作為 Layer 的判斷依據；pointBegin 與 pointEnd 為圖像起始與終點座標，在

每次遞迴先執行 `getBinaryImageValueList()` 並使用 `getMergeColor()` 取得區塊應被填滿的顏色 mergeColor：

如果 mergeColor 為灰色且 treeHeight 不為 0，將 treeHeight – 1、區塊切割成四等分繼續遞迴。

如果 mergeColor 為單色，或者 treeHeight = 0 代表 Quad-tree 的深度以達到目標 Layer 數，呼叫 `updateQuadTreeImage()` 將該區塊塗上 mergeColor 的顏色，並且結束遞迴。

```
void createQuadTreeImage(int layer)
```

呼叫 `updateQuadTreeRecursively()`遞迴，treeHeight 為 Layer 數、pointBegin = (0, 0)、pointEnd = (圖長, 圖寬)，結束所有遞迴後就能取得指定 Layer 數的 Quad-tree image。

1.png

Layer_1

Layer_2

Layer_3

Layer_4

Layer_5

Layer_6

Layer_7

Layer_8

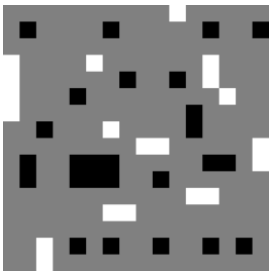
2.png

Layer_1

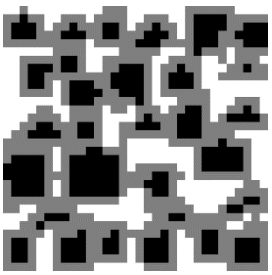
Layer_2

Layer_3

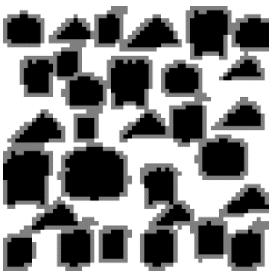
Layer_4



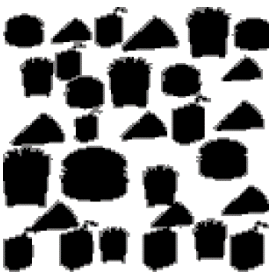
Layer_5



Layer_6



Layer_7



Layer_8

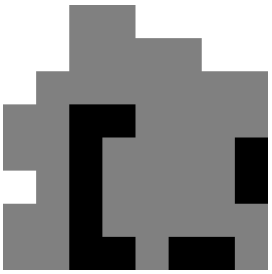
Layer_9

3.png

Layer_1

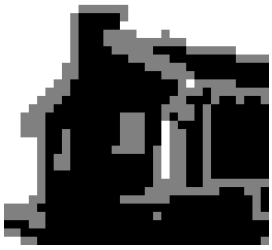
Layer_2

Layer_3



Layer_4

Layer_5



Layer_6

Layer_7

Layer_8

4.png

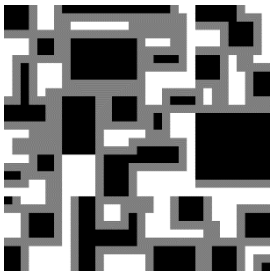
Layer_1

Layer_2

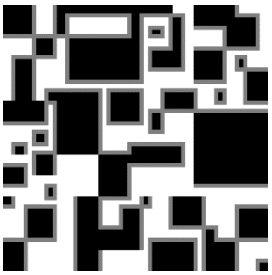
Layer_3

Layer_4

Layer_5

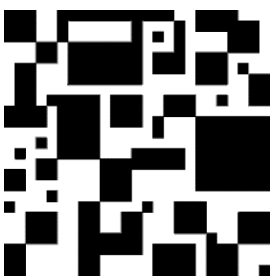


Layer_6



Layer_7

Layer_8



Layer_9