クロマキーによる画像合成

クロマキー ・・・　画像内の各画素の色情報を元に。前景(合成させたい物体)と背景の領域を切り分け最後に各パーツをつなげることで合成画像を作成する方法。

～クロマキーの実装～

今回は以下の画像を使用しクロマキーを実装します。

図1 背景画像
(http://wall.kabegami.com/word/%E6%B5%B7?page=1)

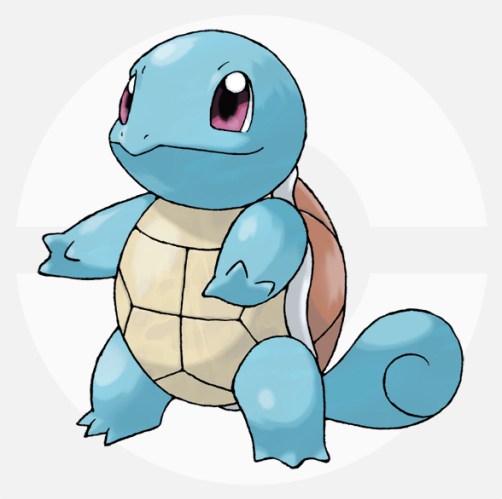

図2 合成対象画像
(https://kokubanlemon.hatenablog.com/entry/2018/12/01/143540)

方法1　物体の抽出

① 閾値240で二値化画像を生成、二値化画像を元にマスク画像を生成。

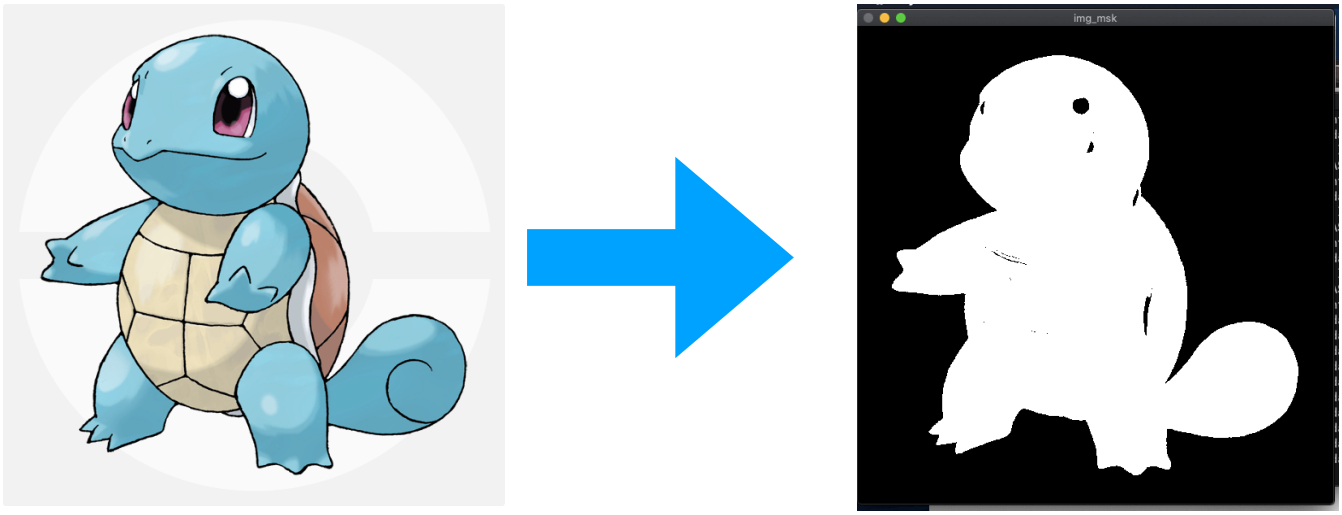

図3 マスク画像の生成

② 元画像からマスク画像の部分だけを切り出す

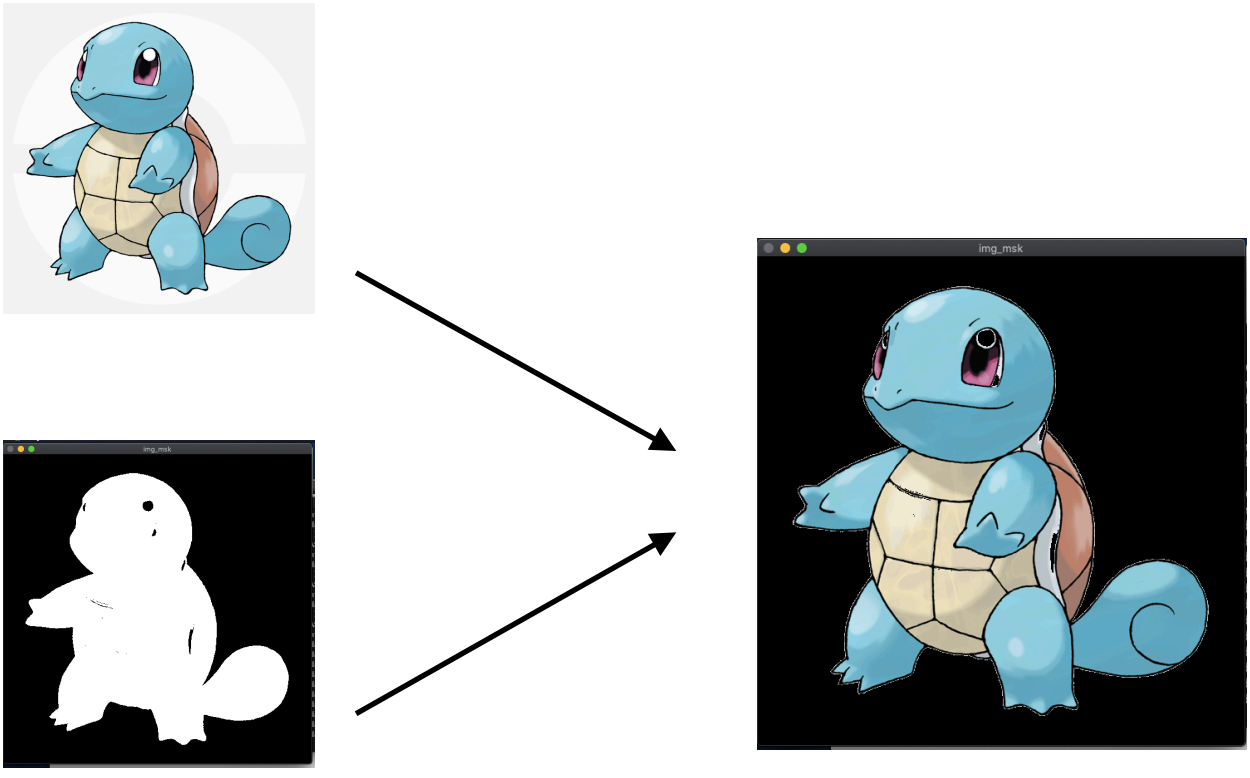

図4 物体の切り出し図解

方法2　背景からの切り出し

① 先ほど使用したマスク画像を反転

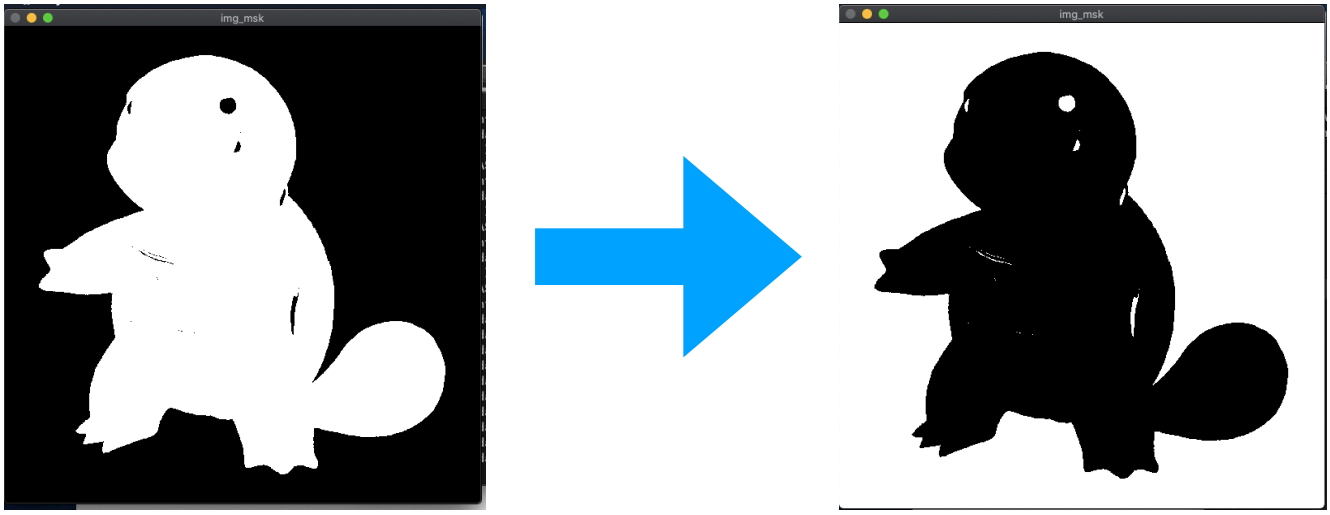

図3 背景用マスク画像の生成

② 背景画像から背景用マスク画像分だけ切り出す

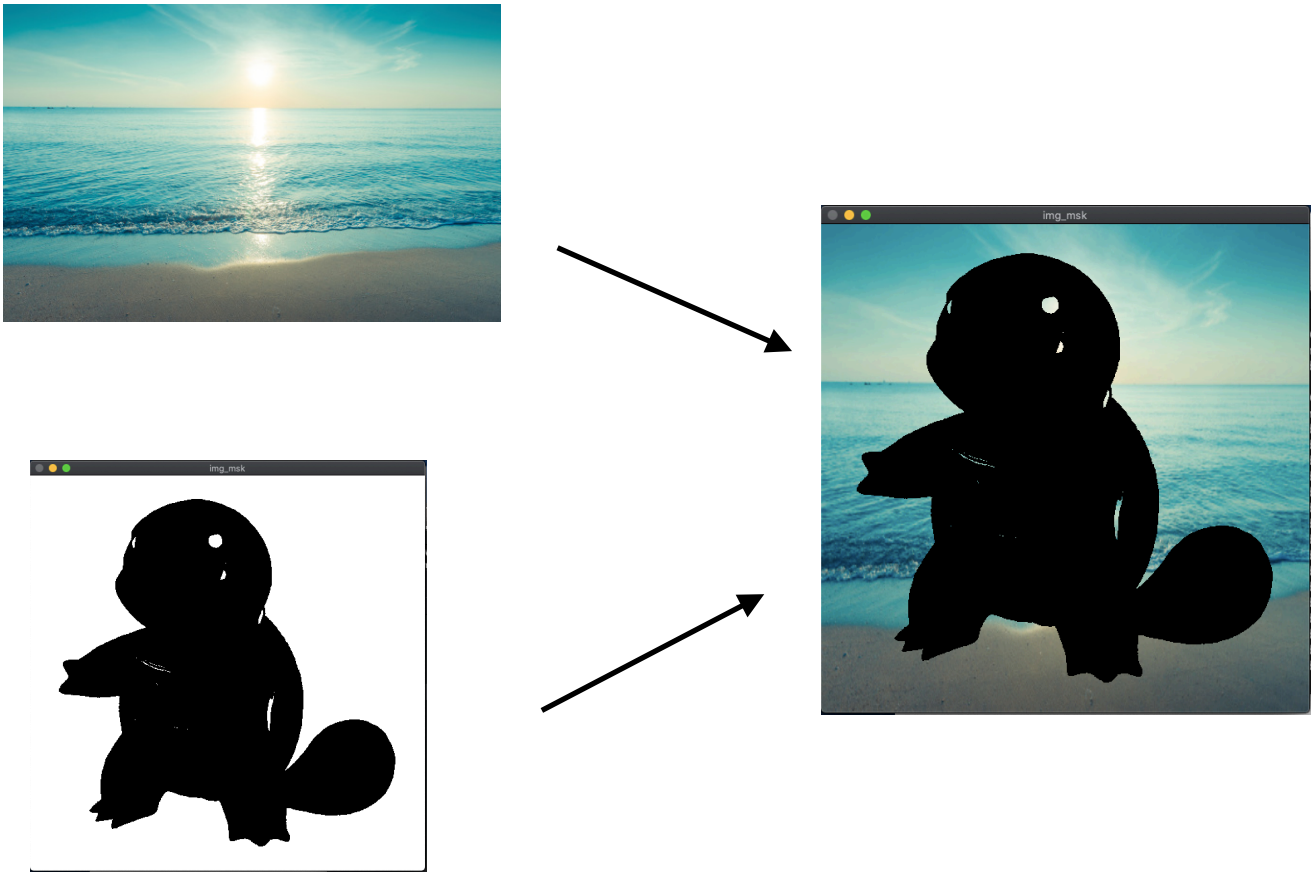

図4 背景からの物体の切り出し図解

方法3　画像の合成

方法1、2から作成した物体のみの画像と物体を入れたい部分を切り抜いた画像を合成

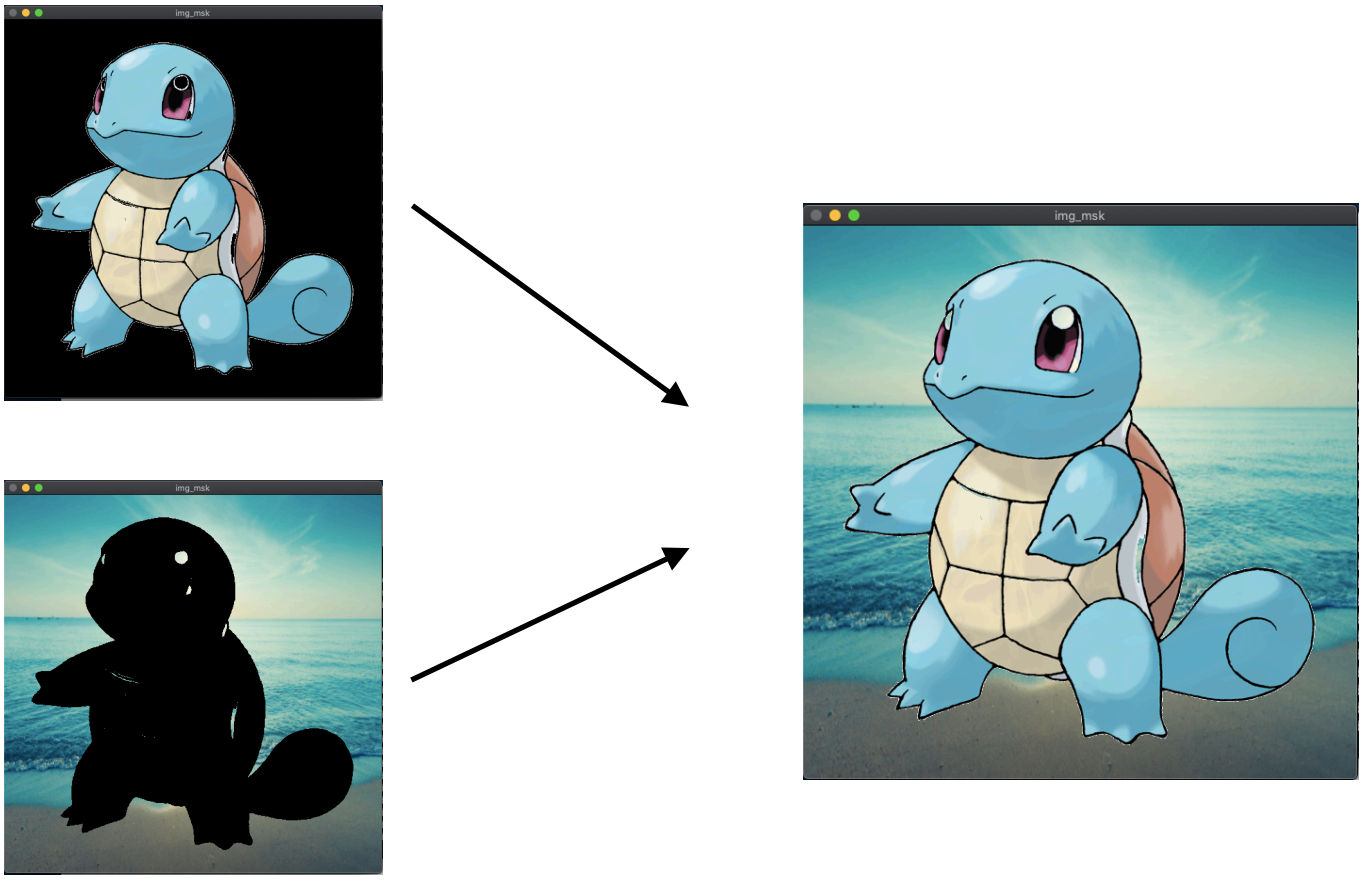

図4 合成部分図解

問題点

　これは二値化を使用するのですが、グレースケール変換した後の合成対象画像の背景と合成対象のピクセル値に差がある(背景が0に近く対象物が255に近いなど)場合は上記のように綺麗な合成画像ができますが、合成対象のピクセル値にあまり差がなくなってしまった場合綺麗な合成画像ができなくなってしまいます。

例)

うまく合成できてない

解決策

上記画像の場合、背景では紫が多く使用されているため、二値化する前に最も多いRGBの値をそれぞれ0に置き換えれば閾値が明確になり綺麗な合成画像を作成できると考えられます。

※どのようにして指定した画素値を持ったピクセルの色を変換については現在リサーチ中です。

簡単ではありますが作成したコードを載せておきます。

```
import cv2

img_src1 = cv2.imread("20181201105601.jpg")
img_src2 = cv2.imread("back.jpg")

#対象物が写っている画像のサイズを取得
height = img_src1.shape[0]
width = img_src1.shape[1]

#画像のサイズを合わせる(今回は引き伸ばしで対応)
img_src2 = cv2.resize(img_src2, (int(width), int(height)))

img_g1 = cv2.cvtColor(img_src1, cv2.COLOR_BGR2GRAY)

img_mskg = cv2.threshold(img_g1, 240, 255, cv2.THRESH_BINARY_INV)[1]

img_msk = cv2.merge((img_mskg, img_mskg, img_mskg))
img_s1m = cv2.bitwise_and(img_src1, img_msk)
img_mskn = cv2.bitwise_not(img_msk)

img_s2m = cv2.bitwise_and(img_src2, img_mskn)
img_dst = cv2.bitwise_or(img_s1m, img_s2m)

cv2.imshow("img_msk", img_dst)
cv2.waitKey(0)
cv2.destroyAllWindows()
```