

中文版 FastTemplateMatching 文档

Copyright: 王肇宁 (Zhaoning(Eric) Wang)

Company: Siasun Robotics 新松机器人

代码 Github 地址: <https://github.com/EricWang12/Siasun-Template-Matching>

开发自 dajuric 的开源库: <https://github.com/dajuric/accord-net-extensions>

注:

所有方法均整合到 TemplateMatching.cs, 其他文件比较原开源 demo 均有部分更改。

如果有条件请去 Github 上读英文版的 README!!!!

使用方法:

NO.1:

使用和安装开源库:

库文件已自带与 package 文件夹, 主要来说使用了三个 Accord 库文件:

Image processing

* __Accord.Extensions.Imaging.Algorithms package__

Implements image processing algorithms as .NET array extensions including the Accord.NET algorithms.

Math libraries

* __Accord.Extensions.Math package__

Fluent matrix extensions. Geometry and graph structures and extensions.

Support libraries

* __Accord.Extensions.Imaging.AForgeInterop package__

Interoperability extensions between .NET array and AForge's UnmanagedImage.

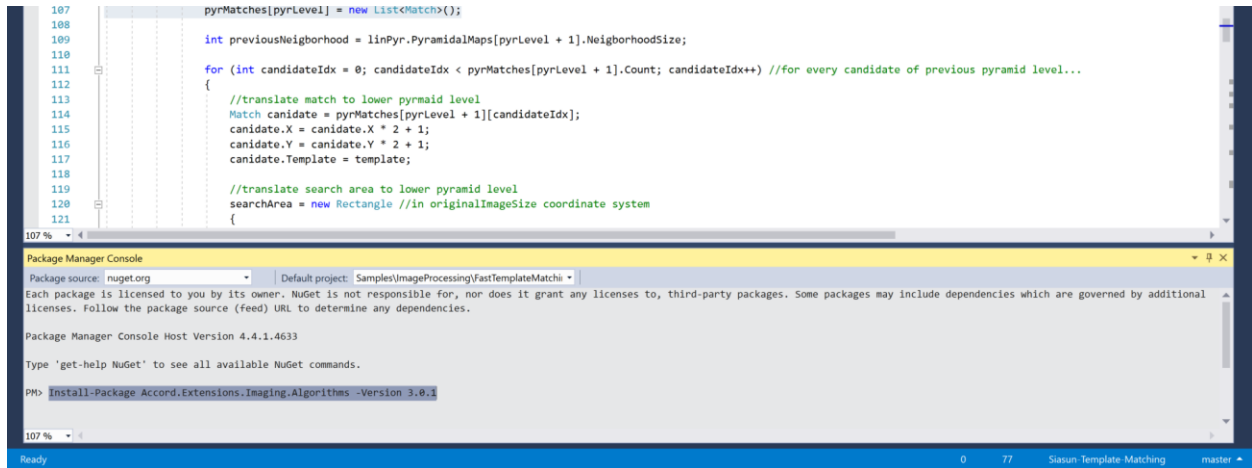
安装:

在 visual studio 里面的 package manager 打入:

```
PM> Install-Package Accord.Extensions.Imaging.Algorithms -Version 3.0.1
```

```
PM> Install-Package Accord.Extensions.Math -Version 3.0.1
```

PM> Install-Package Accord.Extensions.Imaging.AForgeInterop -Version 3.0.1



如果需要其他的库文件资料: <https://www.nuget.org/profiles/dajuric>

-----正文分割线-----

基本方法

在 TemplateMatching.cs 里面有一个 region 叫 “**Basic Methods**”

里面是该文档的通用方法:

[具体变量用途以及用法皆标注于方法注释内]

#1: 建立模板:

- 从一个图像创建模板:

```
public static List<TemplatePyramid> buildTemplate(Gray<byte>[,] image, int Width,
int Height, bool buildXMLTemplateFile = false, int angles = 360, int sizes = 1, float minRatio
= 0.6f, int[] maxFeaturesPerLevel = null)
```

注: 这是从一个 Gray Type 文件创建模板, Gray 是这个项目通用的图像格式 (from Accord Library) 从文件建立模板请参见下一条

- 从文件创建模板:

```
public static List<TemplatePyramid> fromFiles(String[] files, bool
buildXMLTemplateFile = false, int angles = 360, int sizes = 1, bool CropToSqr =
false, int[] maxFeaturesPerLevel = null)
```

输入 String[] 为文件路径

- 或--简易方法:

```
public static void buildTemplate(string[] fileNames, ref List<TemplatePyramid> templPyrs, bool saveToXml = false)
```

- 读取模板文件直接读取已存模板:

```
public static List<TemplatePyramid> fromXML(String fileName)
```

注: XML 文件为之前从文件或图像创建模板时建立。

- 建立单个模板:

如果有需要建立**单个模板**并加入 *TemplateList* :

```
TemplatePyramid newTemp = TemplatePyramid.CreatePyramidFromPreparedBWImage(preparedBWImage, templateName, ImageAngle, maxNumberOfFeaturesPerLevel: maxFeaturesPerLevel);  
templateList.Add(newTemp);
```

-----前后分割线-----

#2: 寻找模板:

- 记录寻找时间:

```
public static List<Match> findObjects(Bgr<byte>[,] image, List<TemplatePyramid> templPyrs, out long preprocessTime, out long matchTime, int Threshold = 80, String[] labels = null, int minDetectionsPerGroup = 0, Func<List<Match>, List<Match>> userFunc = null)
```

- 不记录寻找时间:

```
public static List<Match> findObjects(Bgr<byte>[,] image, List<TemplatePyramid> templPyrs, int Threshold = 80, String[] labels = null, int minDetectionsPerGroup = 0, Func<List<Match>, List<Match>> userFunc = null)
```

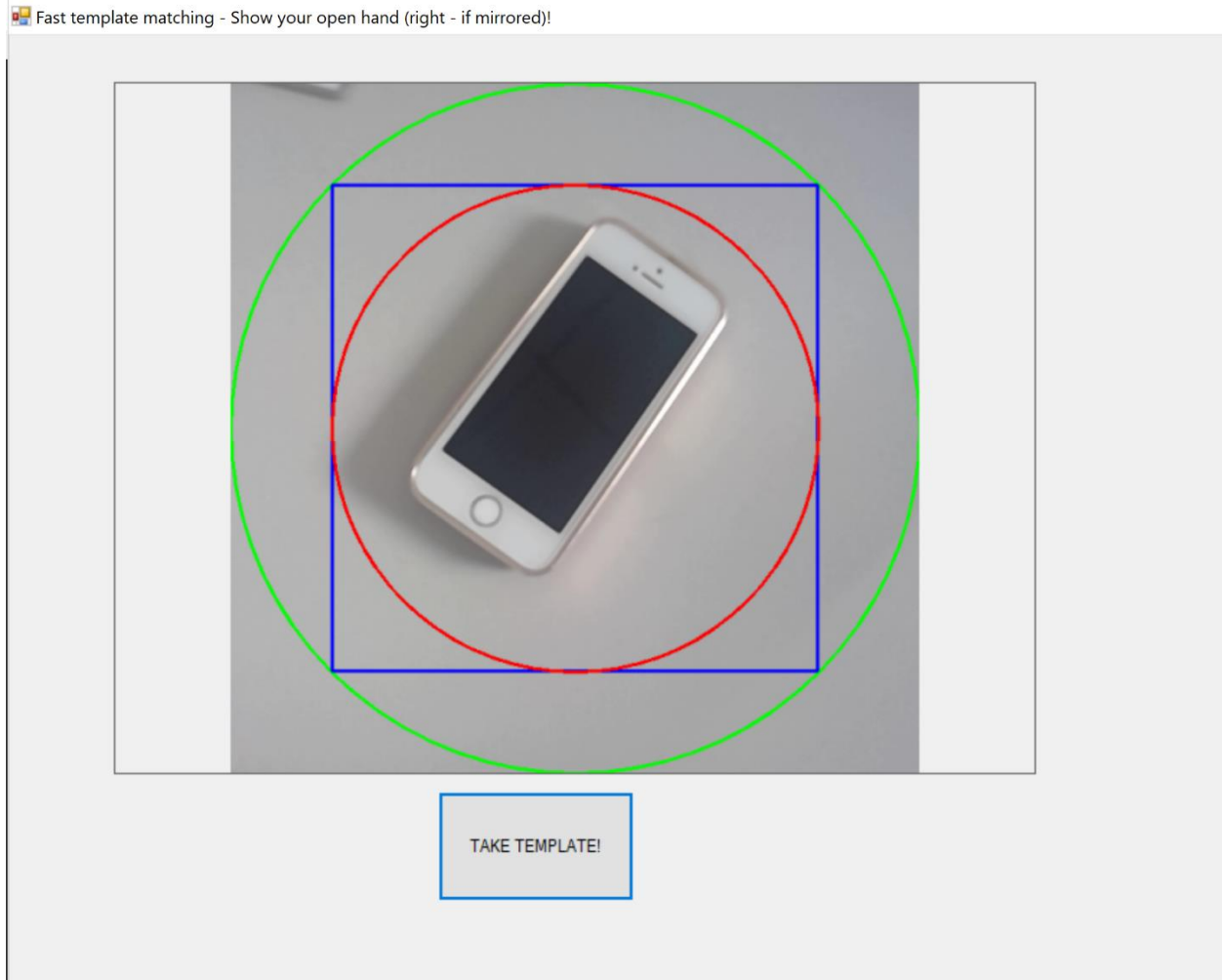
注: findObject 返回一个 Match List 包括当前找到的所有匹配的模板包装成的 match 类, 其中包括位置, 角度, 对应模板等。

-----DEMO 分割线-----

DEMO 流程:

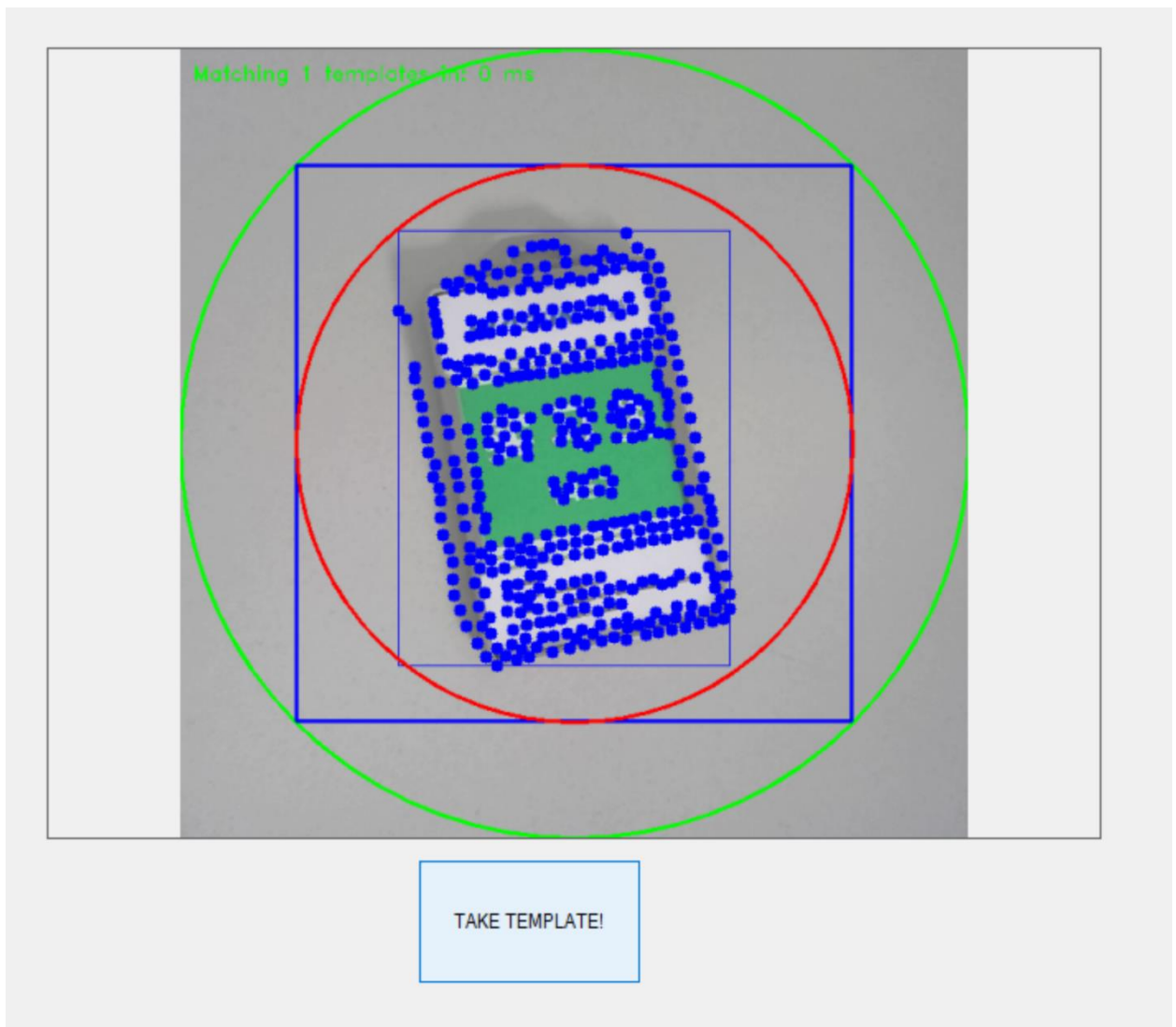
【代码区域请看 在 TemplateMatching.cs 里面还有一个 region 叫 “**build template with Camera---**
DEMO”

开始：把物体放于红圈内，并确保红圈和绿圈中间没有干扰物。



点击 TAKE TEMPLATE!!

注：我加了一个 *validateFeatures* 作为 *userFunc* 的试例，它会过滤掉绿圈以外的任何矢量（蓝点）



一个模板会被制作出来作为预览，在 console 里面确认模板，就会开始制作

可把我自己给牛逼坏了

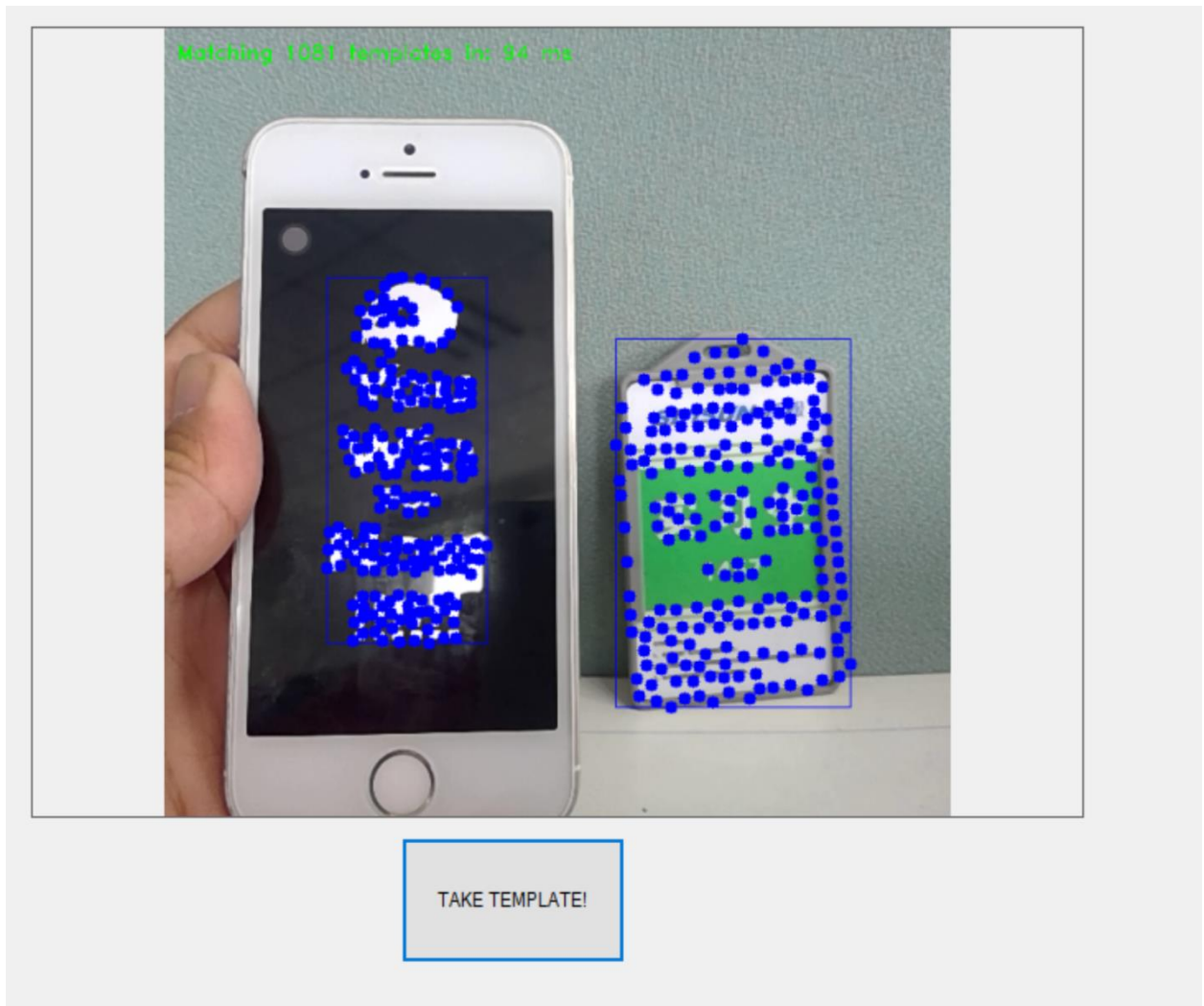


叉会儿腰

成功啦

是否制作另一个模板，y 重复， 其他便进入最后识别环节

一个多个模板同时识别的例子：

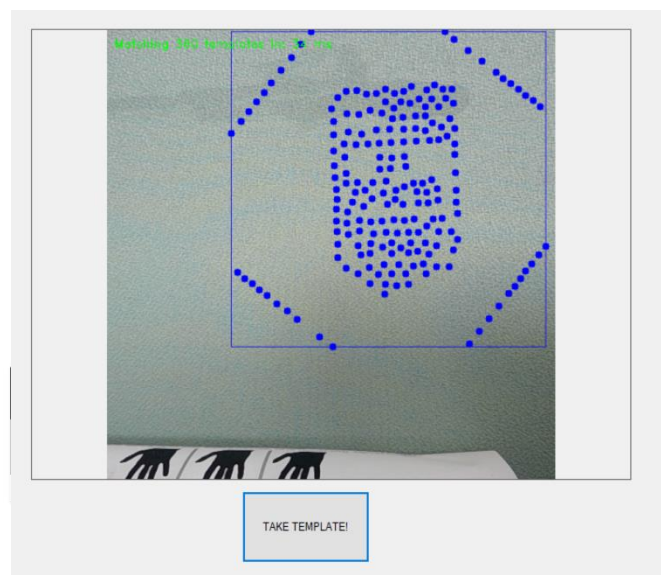
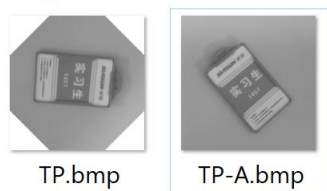


执行过程分析：

在初始阶段，程序会以全部图像生成模板预览，但在制作阶段，会把图像切割成蓝色正方形大小并以此制作模板。最后以制作出来的模板（List）来寻找图像。

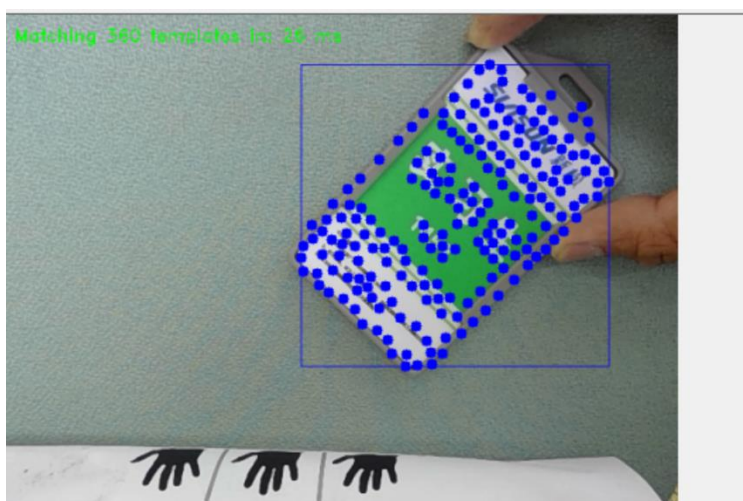
这么做的原因是因为如果以原图为源图像，在旋转时会有非本图的透明色（黑色）加入进来，而算法会把边界看为模板的一部分所以不可取。而蓝色正方形（原图像短边大小的 $\sqrt{2}/2$ ）可以在图像内自由旋转。

所以，除非原图背景为纯黑，其他情况直接用 `fromFile` **可能** 会导致模板不对



因为，fromFiles 方法依旧是旋转源文件，所以上述情况会发生。

解决方法：在 fromFiles 里面的参数 CropToSqr 设置为 true，这样就会把其剪成 sqrside（原图像短边大小的 $\sqrt{2}/2$ ）



如果你有开拓精神请读：

整套系统虽然是 Pyramid structure 为主，但为了系统稳定性只用了一层（相当于没有），如果想使用多层金字塔结构，更改 `DEFAULT_NEGBORHOOD_PER_LEVEL` from `LinearizedMapPyramid.cs : 39` 和 `DEFAULT_MAX_FEATURES_PER_LEVEL` from `ImageTemplatePyramid.cs:56`. 数列的个数即为金字塔的层数（上述两个数列个数必须一致）。

但现在在使用多层金字塔结构时，在 `detector.cs: calculateSimilarityMap` 会出现 **Fatal Execution Engine Error**，恕能力有限无法解决，若解决则检测速度和精度还会有所提升。