

Gesture Recognizer

Content of this document (click to follow each link):

- [Release Notes](#)
- [Example Scene](#)
- [The DrawDetector component](#)
- [Configuration from scratch](#)
- [Script Reference](#)
- [Tips for better recognition results](#)

Developer contact: raphaelmarquesapps@gmail.com

Release Notes

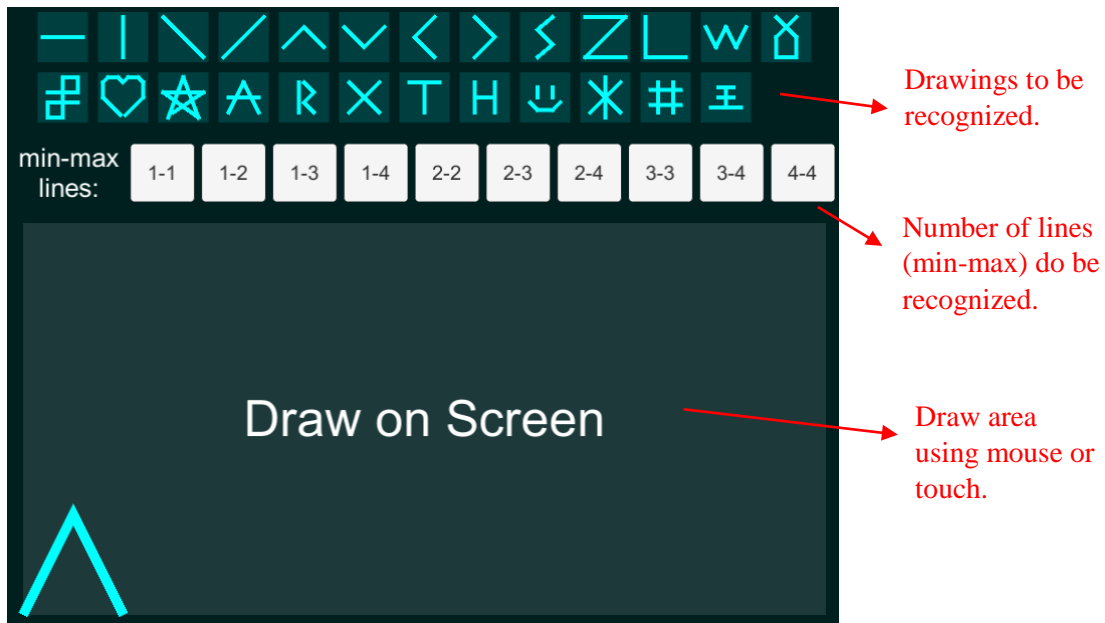
Version 2.0

- Multi Stroke (a gesture made of many lines)
- All scripts inside ***GestureRecognizer*** namespace
- New easier editor do create gestures
- Button do convert old gesture asset to the new one
- Draw area can be placed anywhere on screen

Example Scene

See example scene at ***GestureRecognizer/Example*** folder.

Execute the example scene and draw on bottom area using mouse or touch. The example is set to recognize symbols made of 1 to 4 lines. You can change this configuration clicking on each button. When a symbol is recognized, it blinks and its id shows at the screen.

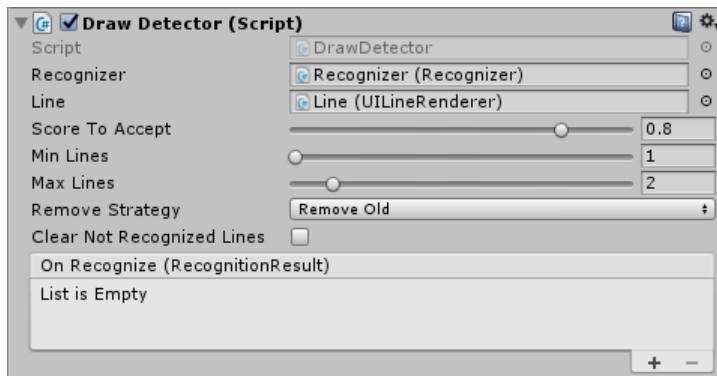


The scene is made of those objects and components:

- Object **"Recognizer"** with **Recognizer** component, filled with **GesturePattern** assets you can recognize.
- Objects **"Canvas/Reference/Image*/Line"** with **GesturePatternDraw** component, with **Pattern** attribute set to draw the **GesturePattern** asset on canvas.
- Objects **"Canvas/Buttons/Button*"** are buttons with **OnClick** event set to change **MinLines** and **MaxLines** attributes of **DrawDetector**.
- Object **"Canvas/DrawArea"** with **DrawDetector** component to handle user drawing on screen, and **ExampleGestureHandler** component, to show how to handle the recognition result.

The DrawDetector component

This component is responsible for capture user drawing (by mouse or touch), store and show what user drew, and recognize it when user stop drawing using the Recognizer component.



Its attributes are:

- Recognizer: the object with Recognizer component attached.
- Line: object inside DrawDetector hierarchy with a UILinerenderer.
- Score To Accept: score (0.0 to 1.0) to accept some drawing.
- Min Lines: minimum number of lines of a gesture.
- Max Lines: maximum number of lines of a gesture.
- Remove Strategy: what to do when the user stop drawing and the number of lines reach the maximum value.
- Clear Not recognized Lines: if will remove lines that aren't part of the recognized gesture.
- OnRecognize: event you can a method to handle the recognition result

Configuration from scratch

To create your own scene you must follow those steps:

1. Create gesture files using the menu **"Assets/Create/GestureRecognizer/GesturePattern"**
 - 1.1. Select the created asset and look to the **Inspector**.
 - 1.2. Configure the **id** attribute. It will be used to know which gesture was recognized.
 - 1.3. Add and remove lines. You draw each line clicking on the grid.
 - 1.4. Check the **Snap** option to draw lines aligned to grid.
 - 1.5. You can create more than one asset with same id. This will give user many ways to draw the same symbol.
2. Create a new object, the **"Recognizer"**
 - 2.1. Add the **Recognizer** component to it.
 - 2.2. Fill the **Patterns** array attribute with all **GesturePattern** assets you want to recognize.
3. Create a canvas image using the menu **"GameObject/UI/Image"**, the **"DrawArea"**
 - 3.1. Add the **DrawDetector** component to it.
 - 3.2. Create a new **"Line"** object inside **"DrawArea"** hierarchy with the **UILineRenderer** component.
 - 3.3. Fill the **Recognizer** attribute of **DrawDetector** with the **"Recognizer"** object.
 - 3.4. Fill the **Line** attribute of **DrawDetector** with the **"Line"** object inside it.
 - 3.5. Fill the **OnRecognize** event call some method to deal with the recognition result.
 - 3.5.1. The method must be public, void, and have a **RecognitionResult** parameter.
 - 3.5.2. See the example script **ExampleGestureHandler**.
4. Make sure there is an **EventSystem** object in your scene. If not, use the menu **"GameObject/UI/EventSystem"** to create it.

Script Reference

- **Class: Recognizer** : MonoBehaviour
 - **Description:** Class to recognize a gesture among many gesture patterns.
 - `public List<GesturePattern> patterns;`
 - List of assets that can be recognized. You can change the list in runtime.
 - `public RecognitionResult Recognize (GestureData data)`
 - Method to find/recognize a gesture (inside data) in the patterns list.
-

- **Class: GesturePattern** : ScriptableObject
 - **Description:** Class to store and edit a gesture
 - `public string id;`
 - The identifier of the gesture, use to know each gesture was recognized.
 - `public GestureData gesture;`
 - The gesture made of many lines.
-

- **Class: GestureData**
 - **Description:** Class to store a gesture.
 - `public List<GestureLine> lines;`
 - List of lines of the gesture.
-

- **Class: GestureLine**
 - **Description:** Class to store one line of a gesture.
 - `public List<Vector2> points;`
 - List of points of the gesture line.
-

- **Class: RecognitionResult**
 - **Description:** Class to hold the recognition result
 - `public GesturePattern gesture;`
 - The gesture found in the Recognizer array of assets.
 - `public Score score;`
 - The score that holds the similarity between two gestures.
-

Tips for better recognition results

- Use pretty different gestures.
- Reduce the number of gestures list.
- Test to find the better “Score To Accept” to your gestures.