

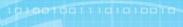
● 骨架追蹤使用了非常複雜的演算法分析深度 影像,並透過轉換矩陣、影像分割與識別、 機器學習等等眾多處理技術,從單純的深度 資料,產生出複雜的骨架與關節點資訊。因 此一般將它稱為骨架追蹤引擎或骨架追蹤系統





- Step 01: 建立新專案-SkeletonState, 在專案中增加對 Kinect SDK 參考
- ✓ Step 02: 配置使用者介面
- Step 03: Window_Loaded_1事件處理常式、 MainWindow()視窗建構函式、 KinectSensors_StatusChanged事件處理常式、UninitalStream函式





❷ Step 04: 資料串流初始化、啟用作業

```
//InitialStream函式
private void InitialStream()
{
    this._myKinect.SkeletonStream.Enable(); //要求Kinect感應器產生資料串流
    //註冊影格備妥事件處理常式
    this._myKinect.SkeletonFrameReady += Kinect_SkeletonFrameReady;
    this._myKinect.Start(); //啟動Kinect感應器硬體
}
```



❷ Step 05: 影格備妥事件處理常式

```
//影格備妥事件處理常式
   private void Kinect_SkeletonFrameReady(object sender, SkeletonFrameReadyEventArgs e)
     Skeleton[] skeletons;
     StateTxt.Text = string.Format("追蹤上限: {0}\n", _myKinect.SkeletonStream.FrameSkeletonArrayLength.ToString());
     using (SkeletonFrame frameData = e.OpenSkeletonFrame()) //取得傳遞的影格資料
       if (frameData == null) //如果影格資料不存在,直接離開事件處理函式
         return;
       //將影格資料複製到資料陣列
       skeletons = new Skeleton[frameData.SkeletonArrayLength];
       frameData.CopySkeletonDataTo(skeletons);
       foreach (Skeleton skeleton in skeletons)
          if (skeleton.TrackingState != SkeletonTrackingState.NotTracked)
           StateTxt.Text += string.Format("追蹤識別碼: {0}\n", skeleton.TrackingId.ToString());
           StateTxt.Text += string.Format("追蹤狀態: {0}\n", skeleton.TrackingState.ToString());
           StateTxt.Text += string.Format("裁剪邊: {0}\n", skeleton.ClippedEdges.ToString());
           StateTxt.Text += string.Format("骨架位置: X: {0}, Y: {1}, Z: {2}\n", skeleton.Position.X.ToString(),
skeleton.Position.Y.ToString(), skeleton.Position.Z.ToString());
           StateTxt.Text += string.Format("-----\n");
```

測試

