# MOG2, KNN, GMG 三种背景减除算法简要对比

## 参数设定

MOG2代码：

|  |
| --- |
| int history = 20;  double varThresh = 0.3;  double learnRate = -1;  bool detectShadows = false;  static Ptr<BackgroundSubtractorMOG2> pMog2 = createBackgroundSubtractorMOG2(history, varThresh, detectShadows);  Mat fgMskMog2;  pMog2->apply(dmat8u, fgMskMog2, learnRate); |

KNN代码：

|  |
| --- |
| history = 20;  double dist2Threshold = 0.8;  pBgSub = createBackgroundSubtractorKNN(history, dist2Threshold, detectShadows);  pBgSub->apply(dmat8u, fgMskMotion, learnRate); |

GMG代码：

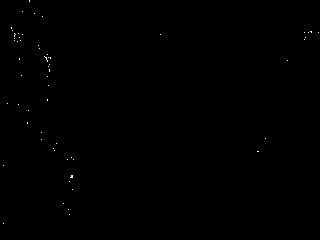
|  |
| --- |
| int initializationFrames = 5,  double decisionThreshold = 0.37;  static Ptr<BackgroundSubtractor> pGmg = bgsegm::createBackgroundSubtractorGMG(initializationFrames, decisionThreshold);  Mat testGmgMat;  pGmg->apply(dmat8u, testGmgMat, 0.03); |

## 单步骤效果

（输入均为8bit灰度图）

KNN调参dist2Threshold:

dist2Threshold = 40;

dist2Threshold = 0.8;

KNN vs. MOG2 vs. GMG (fid=24, 31, 59

KNN:

MOG2:

GMG:

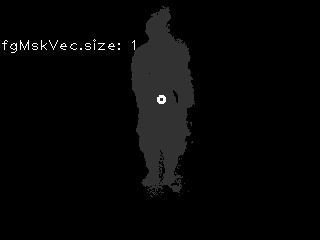
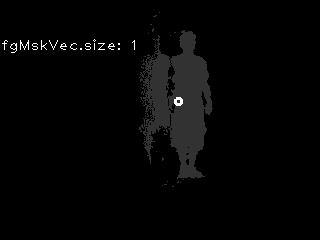
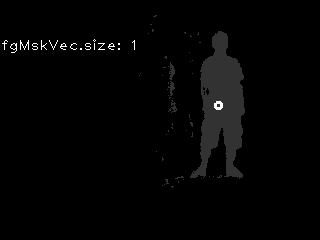
   

可以看出， GMG目前参数设定下效果与KNN/MOG2差距较明显，KNN比MOG2鬼影较小。

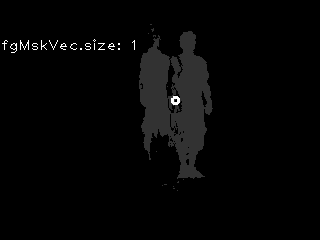
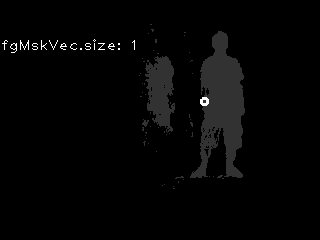
## 整合效果

分别整合 KNN、MOG2到整体流程中，结果对比如下（fid=12, 28, 82, 97）

KNN:

MOG2:

可以看到，KNN比MOG2更容易产生“鬼影”(28帧)；KNN对“鬼影”的消除速度比MOG2更为明显(82, 97帧)。