**1. Source code download**

Requires MS Visual Studio 2010

<https://github.com/xylin/TSB>

**2. Command line usage**

Blood vessel segmentation for fundus image

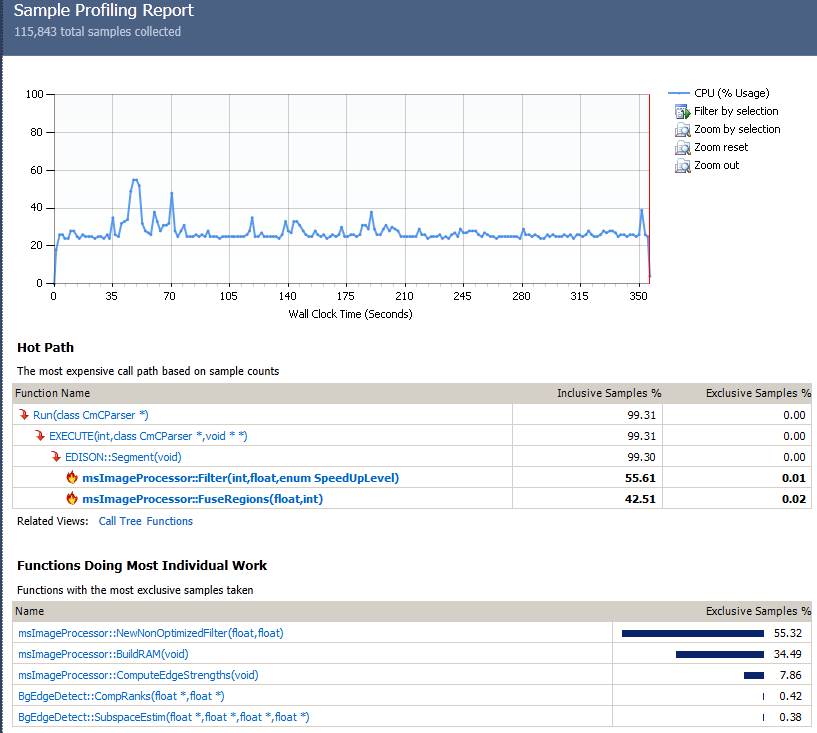
|  |  |  |
| --- | --- | --- |
| tsb.exe | InputFile | OutputFile |

|  |  |
| --- | --- |
| **Parameters**: | * InputFile: input image in jpeg format * OutputFile: output image in jpeg format |

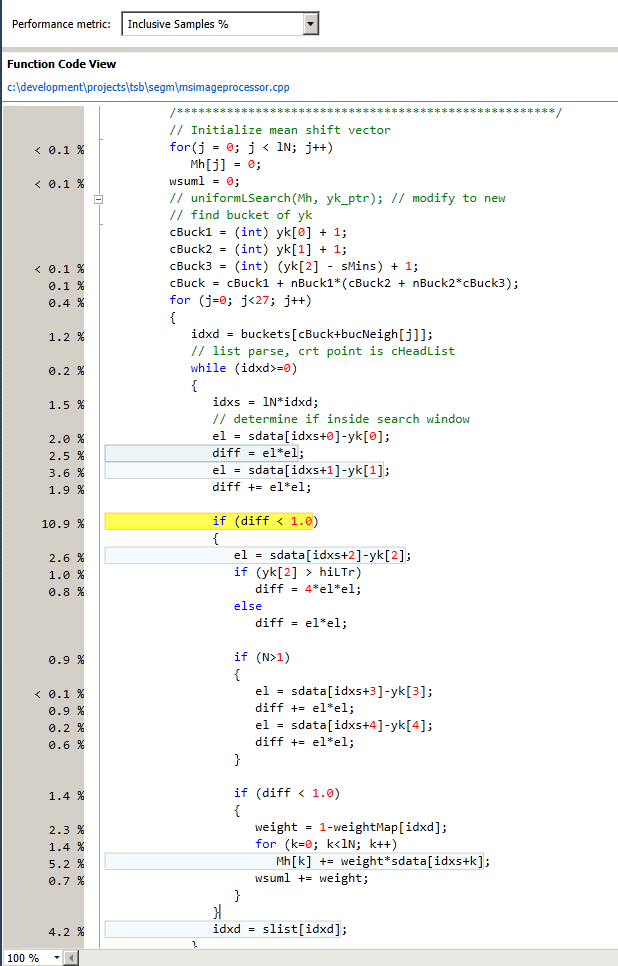
**3. Performance profile**

Figure 1 shows the overview of breakdown on the computation time. The C++ source file required optimisation is msImageProcessing.cpp inside the folder /segm/. The top three functions required GPU speed up are:

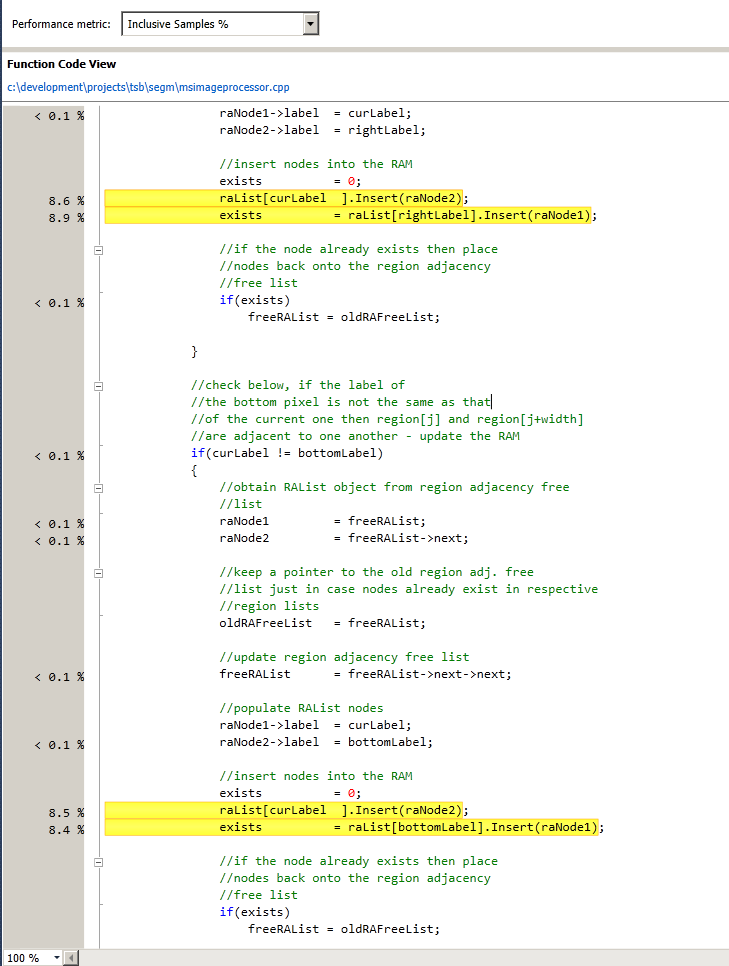
|  |
| --- |
| void msImageProcessor::NewNonOptimizedFilter(float sigmaS, float sigmaR)  void msImageProcessor::BuildRAM( void )  void msImageProcessor::ComputeEdgeStrengths( void ) |



**Figure 1 Profile summery page**



**Figure 2 The most expensive line for NewNonOptimizedFilter**



**Figure 3 The most expensive lines for BuildRAM**



**Figure 4 The most expensive lines for ComputeEdgeStrengths**

**Table 1 The list of for NewNonOptimizedFilter**

|  |  |
| --- | --- |
| **Percentage of processing time%** | **Source Line** |
| 10.89 | 4,586 |
| 5.19 | 4,606 |
| 4.22 | 4,610 |
| 3.55 | 4,583 |
| 2.62 | 4,588 |
| 2.5 | 4,582 |
| 2.27 | 4,604 |
| 1.98 | 4,581 |
| 1.92 | 4,584 |
| 1.49 | 4,501 |
| 1.47 | 4,579 |
| 1.43 | 4,602 |
| 1.42 | 4,605 |
| 1.22 | 4,575 |
| 0.96 | 4,589 |
| 0.94 | 4,594 |
| 0.89 | 4,597 |
| 0.82 | 4,496 |
| 0.8 | 4,590 |
| 0.79 | 4,525 |
| 0.74 | 4,607 |
| 0.72 | 4,521 |
| 0.62 | 4,498 |
| 0.61 | 4,599 |
| 0.52 | 4,616 |
| 0.51 | 4,503 |
| 0.42 | 4,517 |
| 0.38 | 4,573 |
| 0.31 | 4,519 |
| 0.28 | 4,631 |
| 0.27 | 4,520 |
| 0.26 | 4,497 |
| 0.24 | 4,490 |
| 0.22 | 4,499 |
| 0.22 | 4,598 |
| 0.18 | 4,577 |
| 0.15 | 4,504 |
| 0.15 | 4,509 |
| 0.12 | 4,513 |
| 0.1 | 4,488 |
| 0.09 | 4,505 |
| 0.09 | 4,511 |
| 0.08 | 4,368 |
| 0.08 | 4,522 |
| 0.08 | 4,531 |
| 0.05 | 4,492 |
| 0.04 | 4,512 |
| 0.04 | 4,557 |
| 0.04 | 4,564 |
| 0.04 | 4,572 |
| 0.04 | 4,596 |
| 0.03 | 4,613 |
| 0.02 | 4,556 |
| 0.02 | 4,566 |
| 0.02 | 4,630 |
| 0.02 | 4,632 |
| 0.02 | 4,636 |
| 0.01 | 4,365 |
| 0.01 | 4,431 |
| 0.01 | 4,473 |
| 0.01 | 4,479 |
| 0.01 | 4,487 |
| 0.01 | 4,530 |
| 0.01 | 4,543 |
| 0.01 | 4,552 |
| 0.01 | 4,571 |
| 0.01 | 4,615 |
| 0.01 | 4,640 |
| 0.01 | 4,641 |
| 0.01 | 4,645 |
| 0 | 4,406 |
| 0 | 4,407 |
| 0 | 4,409 |
| 0 | 4,430 |
| 0 | 4,433 |
| 0 | 4,434 |
| 0 | 4,436 |
| 0 | 4,471 |
| 0 | 4,472 |
| 0 | 4,481 |
| 0 | 4,514 |
| 0 | 4,528 |
| 0 | 4,542 |
| 0 | 4,654 |

**Table 2 The list of for** BuildRAM

|  |  |
| --- | --- |
| **Percentage of processing time%** | **Source Line** |
| 8.87 | 2,171 |
| 8.64 | 2,170 |
| 8.47 | 2,206 |
| 8.4 | 2,207 |
| 0.03 | 2,138 |
| 0.02 | 2,142 |
| 0.01 | 2,123 |
| 0.01 | 2,141 |
| 0.01 | 2,143 |
| 0.01 | 2,162 |
| 0.01 | 2,202 |
| 0 | 2,101 |
| 0 | 2,121 |
| 0 | 2,122 |
| 0 | 2,149 |
| 0 | 2,153 |
| 0 | 2,154 |
| 0 | 2,165 |
| 0 | 2,176 |
| 0 | 2,185 |
| 0 | 2,189 |
| 0 | 2,190 |
| 0 | 2,198 |
| 0 | 2,251 |
| 0 | 2,270 |
| 0 | 2,298 |
| 0 | 2,299 |