

1.2 更新日志

说明:

1.2 的整体升级规模很庞大

目录

zAI 深度学习内核引擎更新.....	2
程序更新.....	3
建模工具链 z_ai_model 更新.....	5
移出没用的导航条，可以直接看到当前使用的内核引擎版本，红色按钮不随时再出现5	
新增自动显存回收.....	5
新增手动显存回收.....	5
新增图像几何编辑支持：.....	6
新增标签分类过滤器.....	6
新增图像语义分割的训练模块.....	7
新增图像语义分割的数据和标签编辑模块.....	7
新增图像语义分割自动标注功能.....	8
使用 delphi xe 10.3.2 构建.....	8
建模工具链 LocalTrainingServer 更新.....	8
建模工具链 TrainingTool 更新.....	8
Demo 更新.....	9
新增图像语义分割的训练与检测 Demo.....	9
新增多边形填充 demo.....	9
新增几何相交线 Demo.....	10
新增 Learn 的 TLMatrix 表达式 Demo.....	10
新增 zExpression 全面测试和做日常换算使用的 Demo.....	11
新增 zDrawEngine 的彩色文本表达式的 Demo.....	11
新增等距放大 Demo.....	12
新增三角切分算法 demo.....	12
新增使用哈夫变换算法做图片旋转矫正的 demo.....	13
新增 PCA 的使用 demo.....	13
新增 LDA 的降维 Demo.....	14
新增使用文本解析引擎对二进制翻译的 demo.....	14
新增使用文本解析引擎大规模排序 VCL 和 LCL 依赖库的 Demo.....	15
新增像素颜色的分割 Demo.....	15
内核 bug 修复.....	16

zAI 深度学习内核引擎更新

- 内核引擎版本由 1.19 升级至 1.20
- 新增图像语义分割 API:训练+识别完整体系
- 图像语义分割支持 LargeScale: 大规模数据集建模技术, 可训练数百 TB 的数据样本
- 受 Nvidia 三方提供的 SDK 影响, 构建工具集从 VS2015 工具集迁移至 VS2017 工具集
- 跟随 nvidia 升级 cuda 的 api 支持体系升级至 cuda toolkit 10.1 update 1 版本, 该版本主要对近期上市新卡的 tensor 处理器提供优化支持, 深度学习的训练效率更高
- 跟随 nvidia 升级 cudnn 库保持与 cuda toolkit 10.1 update 1 版本保持对应
- 跟随 openblas+eigen 升级, 现有的 OD 效率和效果无变化
- 提供 Cuda_x64,MKL_X64,WindowsX64,WindowsX86, 4 种架构

Cuda 必须 x64, 因为 gpu 要使用和显存同等规模的内存作为交换空间

MKL 可以提供 x86, 但是 MKL 的需要额外的很多 DLL 库, 它们的命名都是同名的, 只能以不同目录来区分, 非常容易搞混, 再说, MKL 也不适合深度学习, 所以 MKL 只提供了一个 X64 的版本

WindowsX64+X86 支持, 这是两个库都是基于 IA32 的 cpu 构建的内核引擎, 它们只能支持 OD 和 VideoTracer, 不适合做深度学习

- 将内核的 Key 验证机制升级至 1.1, 不影响 zpascal.net 用户现有的 key 使用
- 后台验证服务器系统同步支持 1.20, 并且仍然兼容 1.19 与之前的版本
- 受华为 5G 风波影响, 在 1.20 版本开发期间, 作者研究过 zAI 对 IOT 支持 (不是 ARM, 而是直接使用 GPU 芯片)。结果: IOT 目前无法满足 AI 的计算量, 5G 时代, 我们可以选用云后台+5G 高带宽优势来解决 AI 应用问题

程序更新

zAI_Editor_Common 是 1.20 新增建模工具链的数据结构库:可以直接读取.AISet 的数据结构,支持跨平台,我们在做大数据提炼时,如果需要导入数据到工具链可以使用它

zAI_Common 数据集支持库,新增新支持图像语义分割使用的分类颜色池

zAI_Common 数据集支持库,新增新支持图像语义分割使用的描述分类外形类

zAI_Common 数据集支持库,声明部分代码太多了,新增 Region 折叠

zAI 引擎支持库,新增图像语义分割支持

zAI 引擎支持库,RunTrainingTask 函数同步支持图像语义分割

zAI 引擎支持库,同步支持图像语义分割的自动标注

zAI 引擎支持库,声明部分代码太多了,新增 Region 折叠

zAI_TrainingTask 模型自动化训练支持库,同步支持图像语义分割,包括预检测数据集,训练结果检测,训练中断还原

zAI_KeyIO 引擎 License 远程验证库,同步支持内核引擎的 1.1 验证机制

zAI_KeyIO 引擎 License 远程验证库,同步支持 1.2 后台验证

Geometry2Dunit 二维几何库,新增支持三角形切分系统

Geometry2Dunit 二维几何库,将原有单一 Vertex 结构 Sampler+render,拆分成两个三角数据结构

Geometry2Dunit 二维几何库,新增 T2DpolygonGraph 支持,我们在 z_ai_model 几何编辑器看到的包围+轮廓机制就是的 T2DpolygonGraph

Geometry2Dunit 二维几何库,将原有的 TPolY 更名为更恰当的命名 TdeflectionPolygon,并且保持 TpolY 的申明

Geometry2Dunit 二维几何库,新增一套三角计算和打包函数

Geometry3Dunit 三维几何库,移除无用的数据结构,新增像素色差计算方法

MemoryRaster 内存光栅库,新增几何填充 API 体系

MemoryRaster 内存光栅库,新增对图像语义分割支持的色彩分割系统

MemoryRaster 内存光栅库,将 RasterColor 这类长命名改为 Rcolor

MemoryRaster 内存光栅库,优化三角投影算法,不再使用 Sampler+Render 一体化的数据结构,而是将它们拆分成 2 个数据来处理

MemoryRaster 内存光栅库,新增包围线支持

MemoryRaster 内存光栅库,新增支持 T2DpolygonGraph 类型的绘制与填充

MemoryRaster 内存光栅库,内部会区分含 0 的整数计数规则来搭配浮点尺度计算

MemoryRaster 内存光栅库,由于走样变小,重新做了一次内置字体,使用更细的光栅字体作为内置字体库

zDrawEngine 绘图引擎库,新增以脚本描述文本绘制过程的功能

zDrawEngine 绘图引擎库,新增虚线绘制支持

zDrawEngine 绘图引擎库，新增 Polygon 类型的绘制与填充支持

zDrawEngine 绘图引擎库，新增 T2DpolygonGraph 类型支持

zDrawEngine 绘图引擎库，新增三角绘制与填充支持

zDrawEngine 绘图引擎库，将 DrawTexture 这类命名更改成了 DrawPicture

zDrawEngine 绘图引擎库，声明部分代码太多了，新增 Region 折叠

zDrawEngine 绘图引擎库，FMX 的支持库子库，zDrawEngineInterface_SlowFMX 可以对 FMX 的 Tbitmap 进行输出，简单来说，zDrawEngine 可以直接输出到 Tbitmap 中

zDrawEngine 绘图引擎库，FMX 的支持库子库，zDrawEngineInterface_FMX 可以对 FMX 的 Tbitmap 进行输出，简单来说，zDrawEngine 可以直接输出到 Tbitmap 中

TextParsing 文本语法表达式解析库，新增矩阵和向量的语法解析

TextParsing 文本语法表达式解析库，新增对 zDrawEngine 的文本绘制过程描述解析

zExpression 句法表达式支持库，新增向量和矩阵值的计算和书写

Learn 统计学支持库，新增 TLmatrix 的数据描述语言

Learn 统计学支持库，新增 TLVec 的数据描述语言

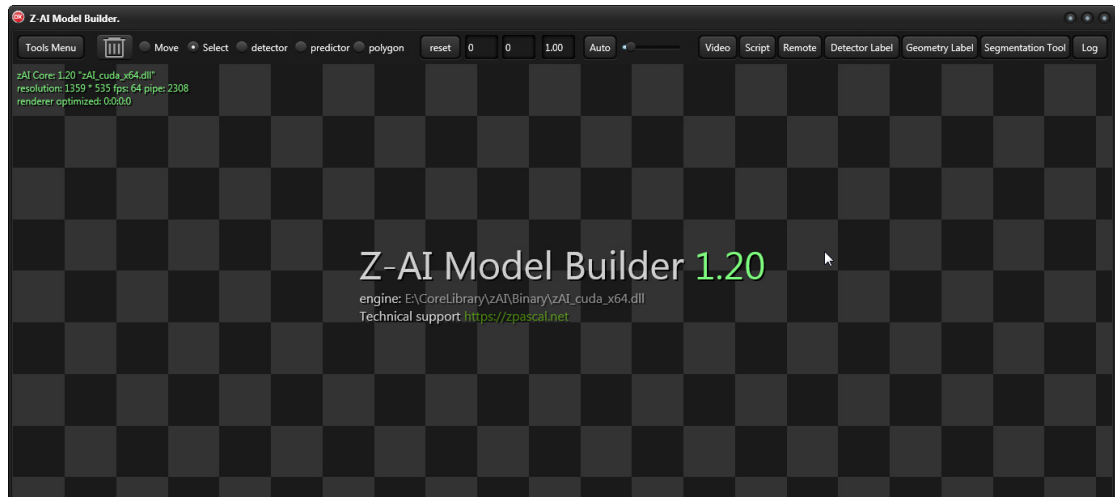
Learn 统计学支持库，小幅优化 PCA 和 LDA 的支持

UnicodeMixedLib 字符串等杂项支持库，小幅新增新的 API

其它库小幅更新若干命名，并无太大改动

建模工具链 z_ai_model 更新

移出没用的导航条，可以直接看到当前使用的内核引擎版本，红色按钮不随时再出现

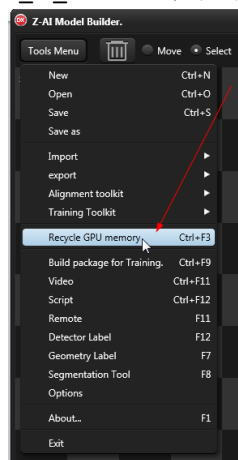


新增自动显存回收

z_ai_model 可以自动化回收显存，当我们有 10000 张图片样本在编辑时，在编辑区看不到的图在 5 秒后会被自动化回收

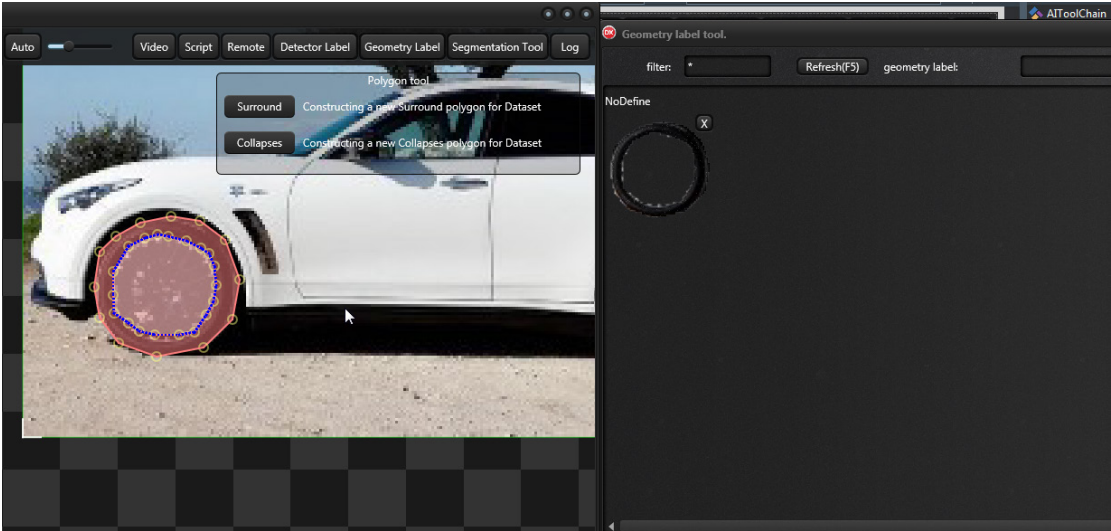
新增手动显存回收

它会回收全部显存，曾经我们在训练模型前，为了让 GPU 达到最高效率，往往直接关闭 z_ai_model，现在不用了，使用热键 **ctrl+f3** 即可达到这个目的



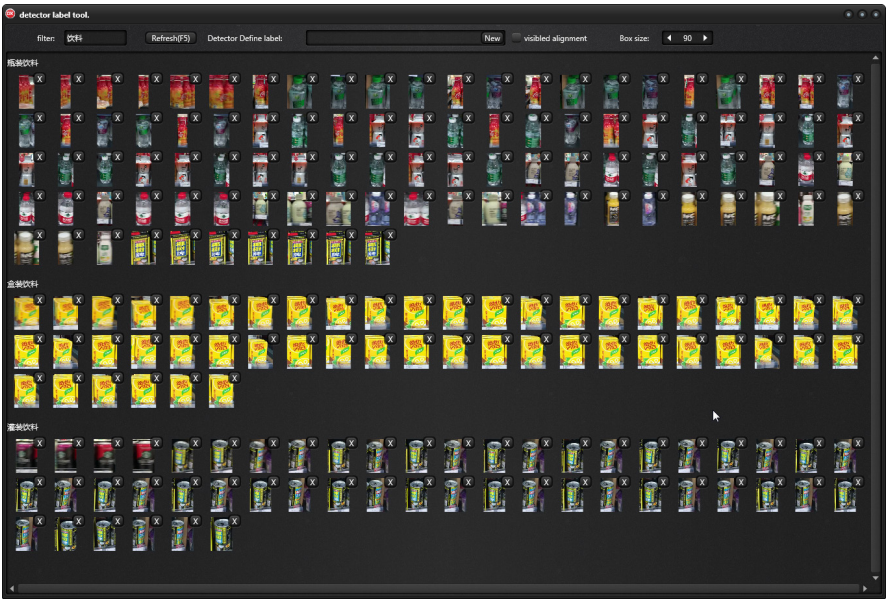
新增图像几何编辑支持:

几何系统的由包围多边形+塌陷多边形共同组成，包围表示平面图像的外部轮廓，塌陷表示在平面图像中被掏空的部分。通过搭配可以用多边形进行任何图像描述，完成深度学习的数据样本集



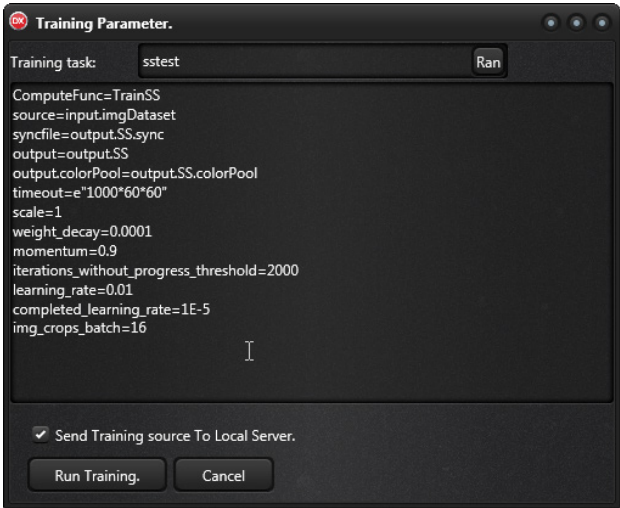
新增标签分类过滤器

ZAI 的开发工艺系统并不赞成在单个数据集中做大量分类,而是让你使用 ZAI_IMGMatrix_Tool 来制作数据集的集合。但是单个数据集的分类多了总是很难维护的，下图是对超市中的数百种分类中过滤出饮料类的结果



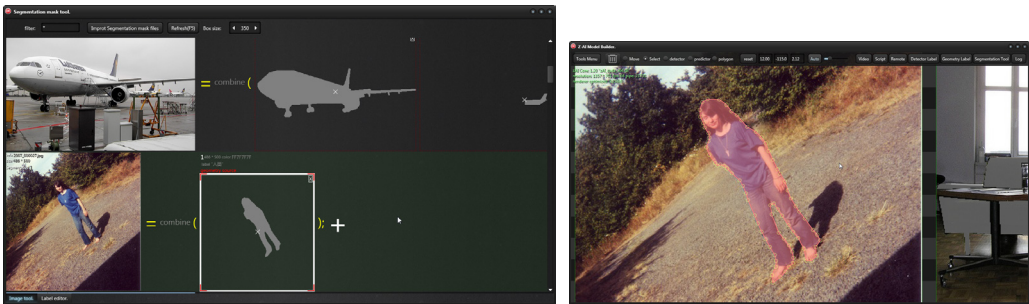
新增图像语义分割的训练模块

从下图我们可以看出图像语义的超参数并不复杂，它对 GPU 的消耗是小型超算级别的。
注意：图像语义分割更侧重于数据样本的构建的编辑，随后我会撰写它的建模指南文档。

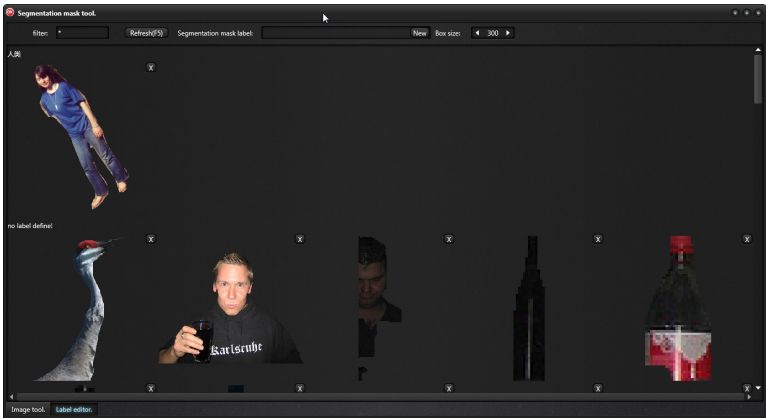


新增图像语义分割的数据和标签编辑模块

图像编辑模块支持几何标签和三方大数据分割和导入，尤其大数据支持，请参考建模指南



下图是分类，它与几何标，检测器标签的概念是一回事



新增图像语义分割自动标注功能

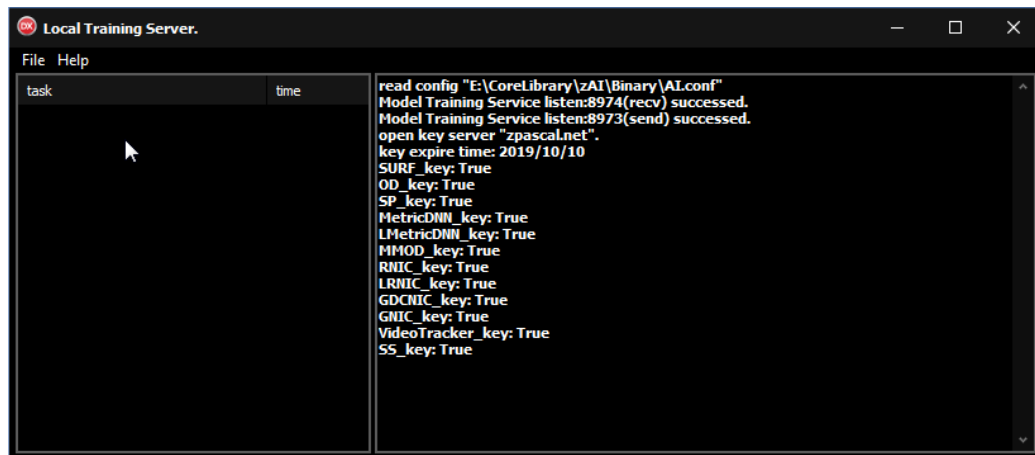
性质与对象检测自动标注类似，它能自动从已训练好的模型中标注人物，车辆，动物的轮廓，帮助我们构建大数据集

使用 delphi xe 10.3.2 构建

10.3.2 修复了很多 bug，对于 zAI 的团队授权用户，我都建议使用这个版本来编译 z_ai_model

建模工具链 LocalTrainingServer 更新

- 同步更新支持图像语义分割训练
- 在右方新增一个任务列表，可以在这里直接 kill task



建模工具链 TrainingTool 更新

- 同步更新支持图像语义分割训练

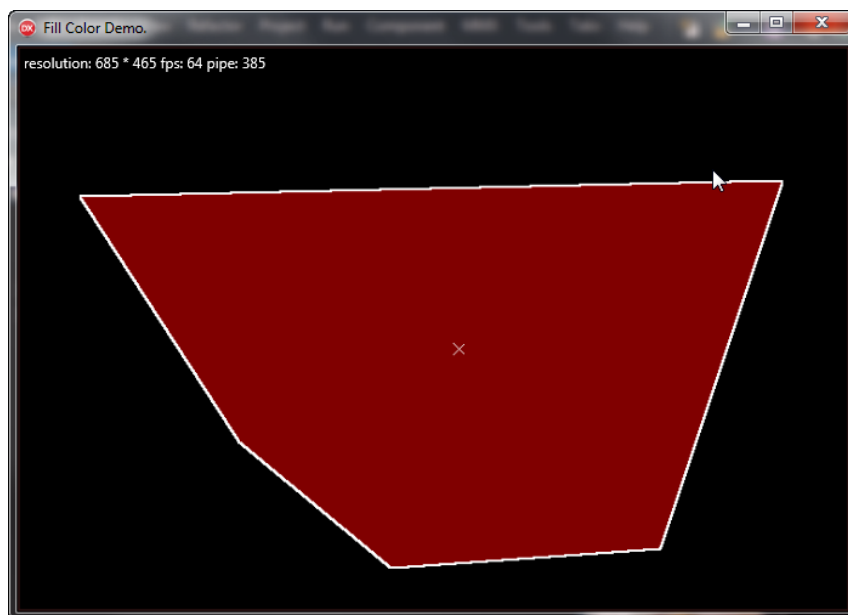
Demo 更新

新增图像语义分割的训练与检测 Demo

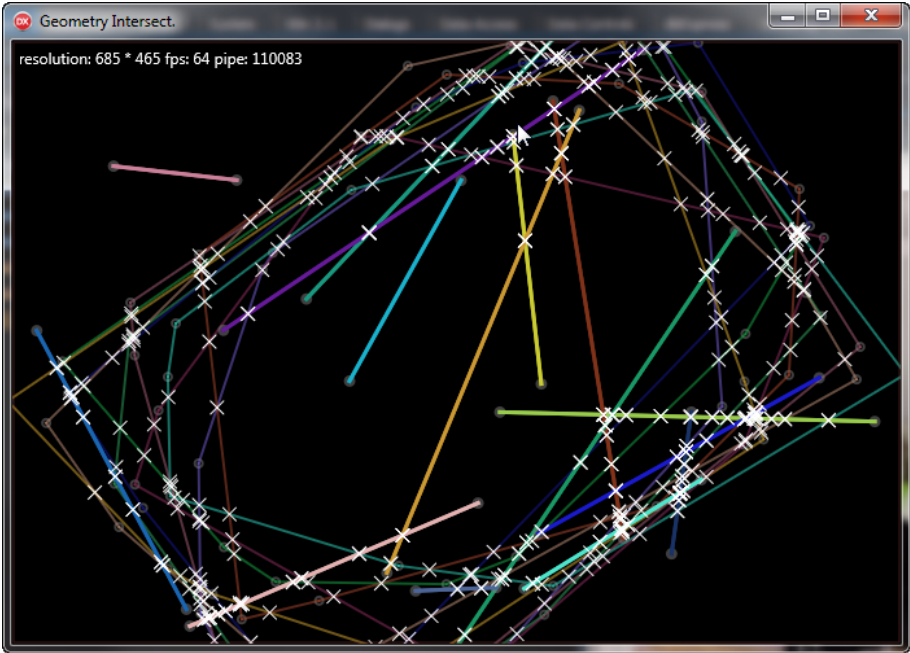
已经充分说明怎样使用图像语义分割 API 进行模型训练和检测



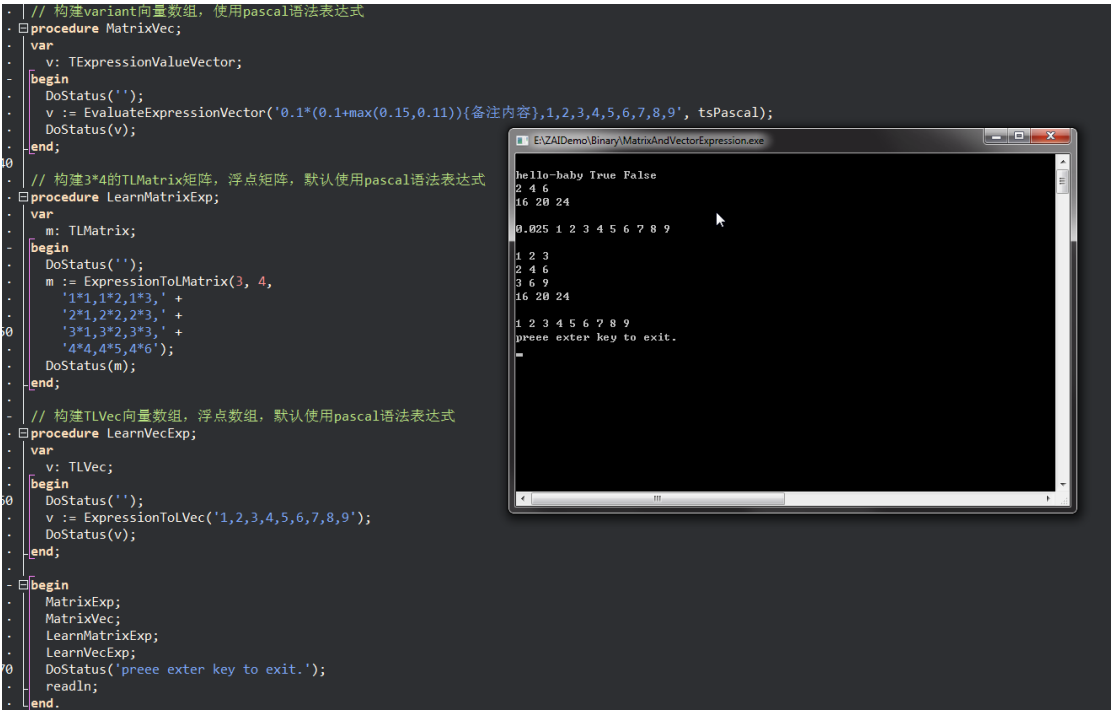
新增多边形填充 demo



新增几何相交线 Demo

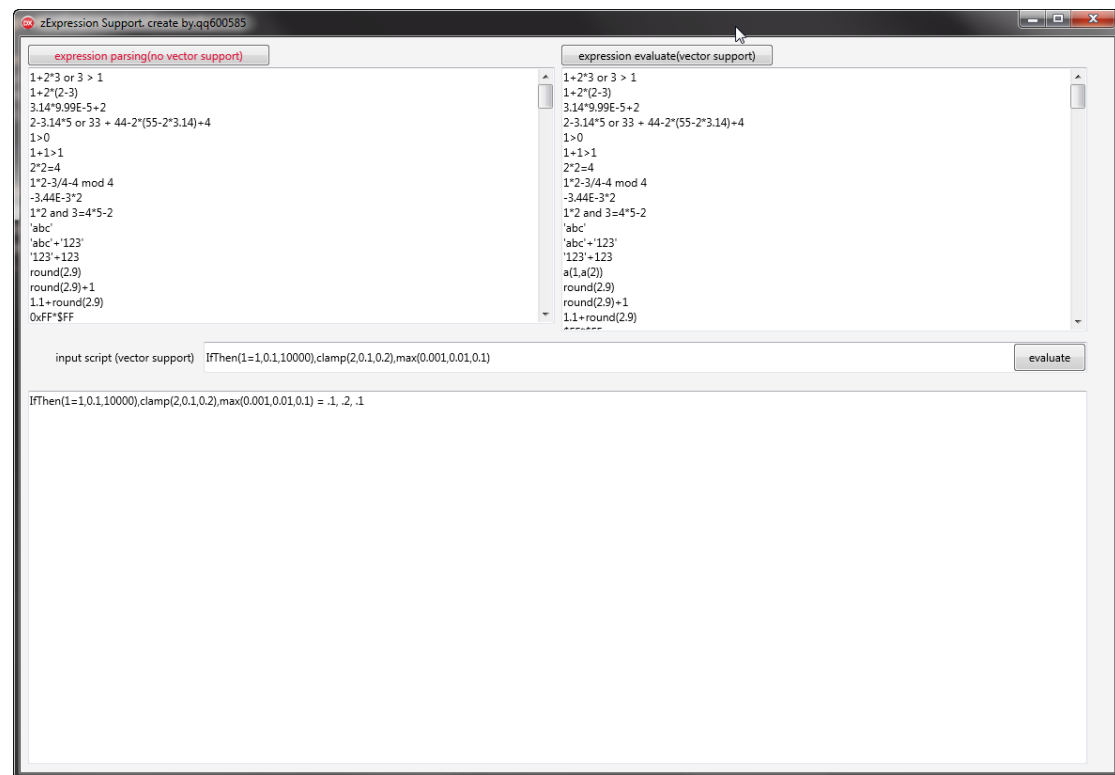


新增 Learn 的 TLMatrix 表达式 Demo

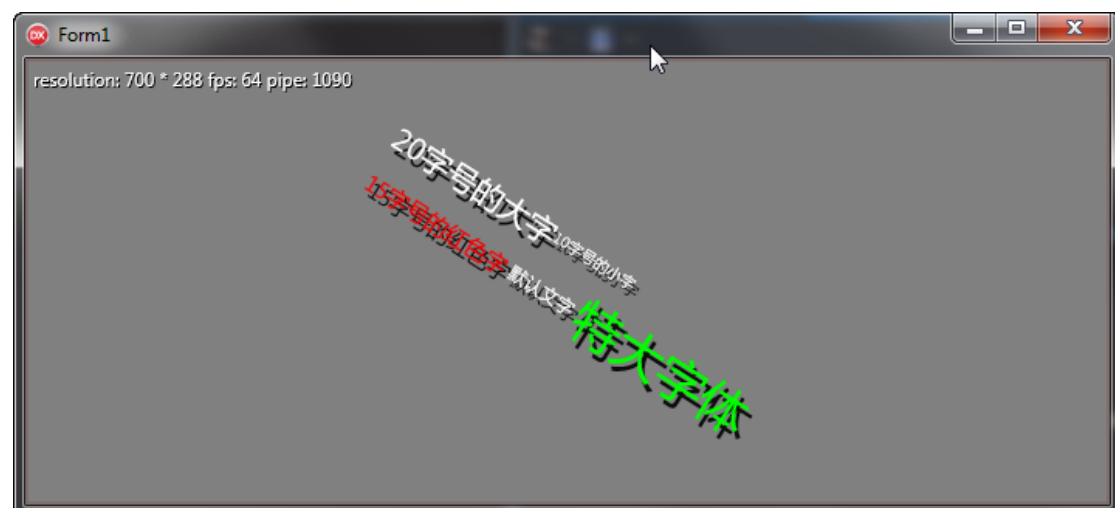


新增 zExpression 全面测试和做日常换算使用的 Demo

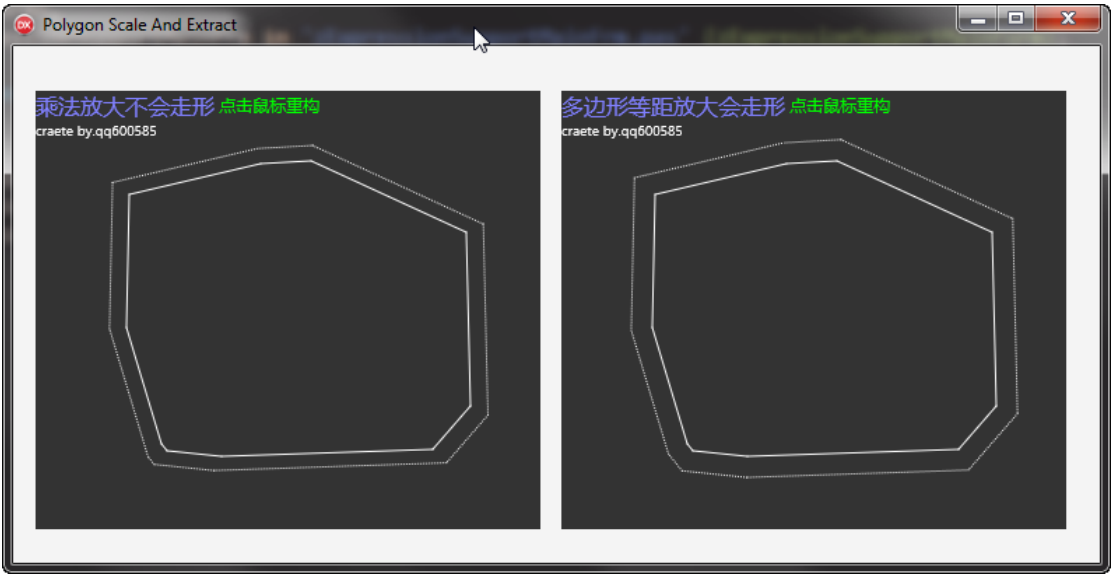
下图我们可以看到它对向量值表达式的换算和处理



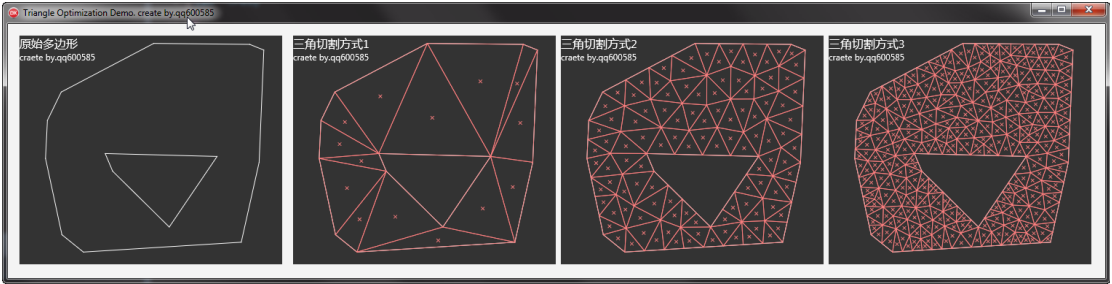
新增 zDrawEngine 的彩色文本表达式的 Demo



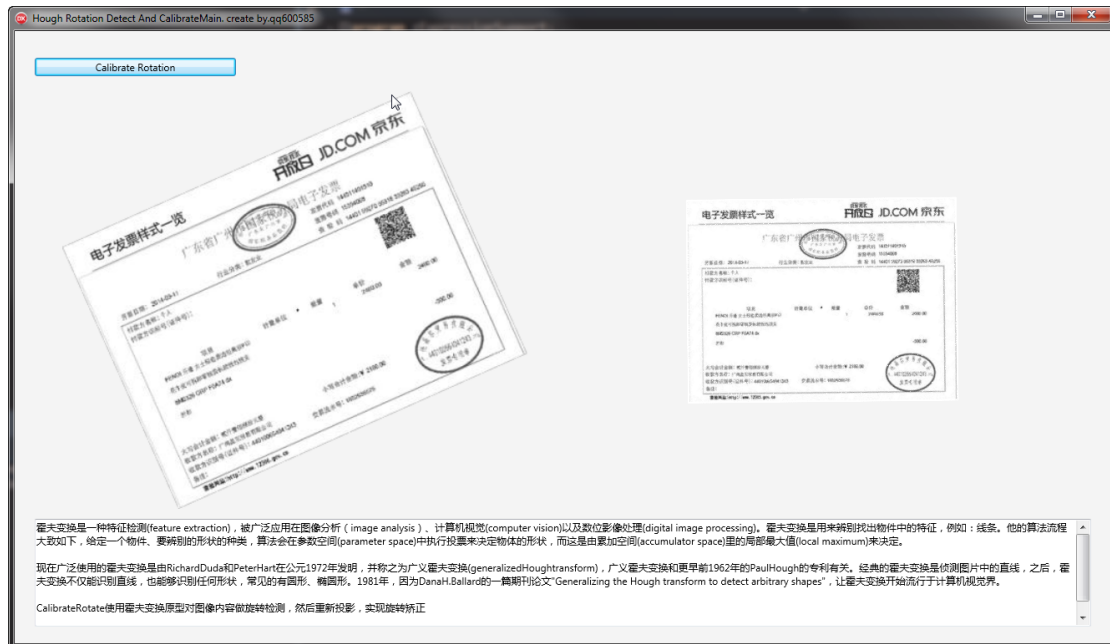
新增等距放大 Demo



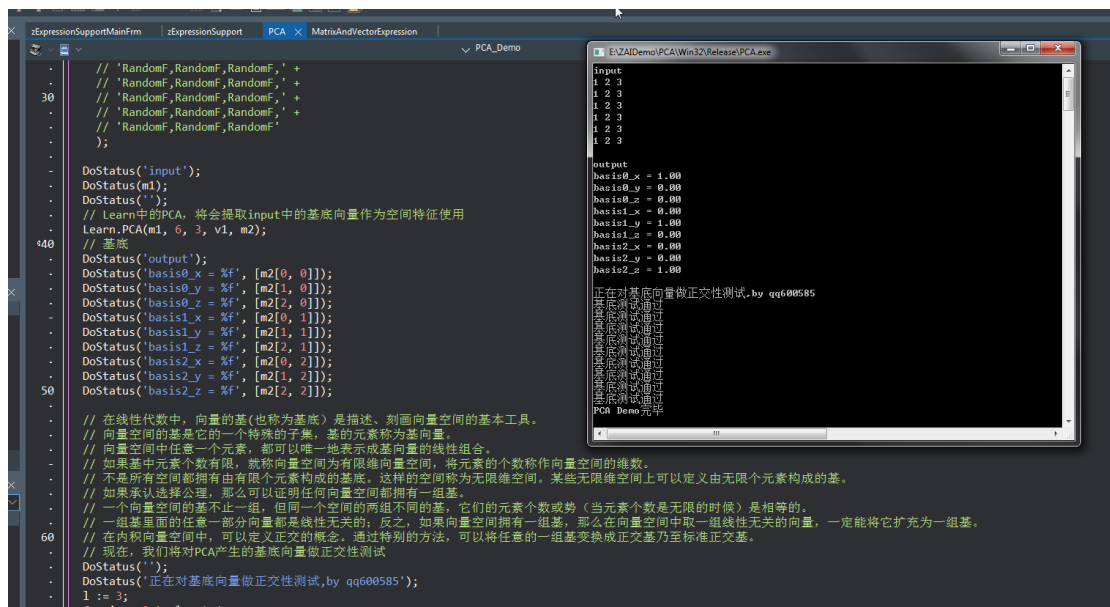
新增三角切分算法 demo



新增使用哈夫变换算法做图片旋转矫正的 demo



新增 PCA 的使用 demo



新增 LDA 的降维 Demo

```
uses SysUtils, PascalStrings, UnicodeMixedLib, DoStatusIO, Learn, LearnTypes;

// 线性判别分析(Linear Discriminant Analysis, LDA), 也叫做Fisher线性判别(Fisher Linear Discriminant, FLD), 是模式识别的经典算法
// 它是在1996年由Belhumeur引入模式识别和人工智能领域的。

// 线性判别分析的基本思想是将高维的模式样本投影到最佳鉴别向量空间, 以达到抽取分类信息
// 投影后保证模式样本在新的子空间有最大的类间距离和最小的类内距离, 即模式在该空间中有
// 因此, 它是一种有效的特征抽取方法。
// 使用这种方法能够保证投影后模式样本的类间散布矩阵最大, 并且同时类内散布矩阵最小。
// 就是说, 它能够保证投影后模式样本在新的空间中有最小的类内距离和最大的类间距离, 即模

procedure LDA_Demo;
const
  sampler_num = 5;
  Classifier_num = 50;
var
  M: TLMatrix;
  cv, v: TLVec;
  i, j: TLInt;
  Info: SystemString;
begin
  // 5个数据样本
  M := UMatrix(Classifier_num, sampler_num);
  // cv是向量形式的分类标签, 由于需要指定分类标签, 所以LDA也被定位成一种有监督的数据
  cv := LVec(Classifier_num);
  for j := 0 to length(M) - 1 do
    begin
      for i := 0 to length(M[j]) - 1 do
        M[j, i] := Random;
        // 分类标签输入类型为整数, 最大值不能超过 Classifier_num
        cv[j] := j + 1;
      end;
    end;

  DoStatus('input');
  DoStatus(M);

  // v是降维后的输出向量, 长度=数据样本
  // 在同纬度空间中v符合线性规律, 既同类型标签
  Learn.LDA(M, cv, Classifier_num, Info, v);
  DoStatus(Info);
  DoStatus(v);
end;

begin
  LDA_Demo;
  main;
end;
```

新增使用文本解析引擎对二进制翻译的 demo

```
number transform. create by.qq600585

const
  SInvalidFileFormat: SystemString = 'Invalid file format';

  {-Blowfish lookup tables}

  bf_P: array [0 .. (BFRounds + 1)] of DWord = (
    $243F6A88, $85A308D3, $13198A2E, $03707344,
    $A4093822, $299F31D0, $082EFA98, $EC4E6C89,
    $452821E6, $38D01377, $BE5466CF, $34E90C6C,
    $C0AC2967, $C97C50DD, $3F84D5B5, $5470917F,
    $92160SD9, $8979FB1B);

  const
    tab_coef_num: array[0..3, 0..16, 0..3] of vlc_bits_len =
    (
      (
        (%1, 1), (%0, 1), (%0, 1), (%0, 1),
        (%000101, 6), (%01, 2), (%00, 1), (%0, 1),
        (%00000111, 8), (%000100, 6), (%001, 3),
        (%000000111, 9), (%00000110, 8), (%00000101, 7),
        (%0000000111, 10), (%000000110, 9), (%000000101, 8),
        (%00000000111, 11), (%00000000110, 10), (%00000000101, 9),
        (%000000000111, 13), (%000000000110, 11), (%000000000101, 13),
        (%0000000000101, 13), (%0000000000110, 13), (%0000000000101, 13)
      )
    );

  const
    SInvalidFileFormat: SystemString = 'Invalid file format';

  {-Blowfish lookup tables}

  bf_P: array [0 .. (BFRounds + 1)] of DWord = (
    608135816, 2242054355, 320440878, 57701188,
    2752067618, 698298832, 137296536, 3964562569,
    1160258022, 9531601567, 3193202383, 887688300,
    3232508343, 3380367581, 1065670069, 3041331479,
    2450970073, 2306472731);

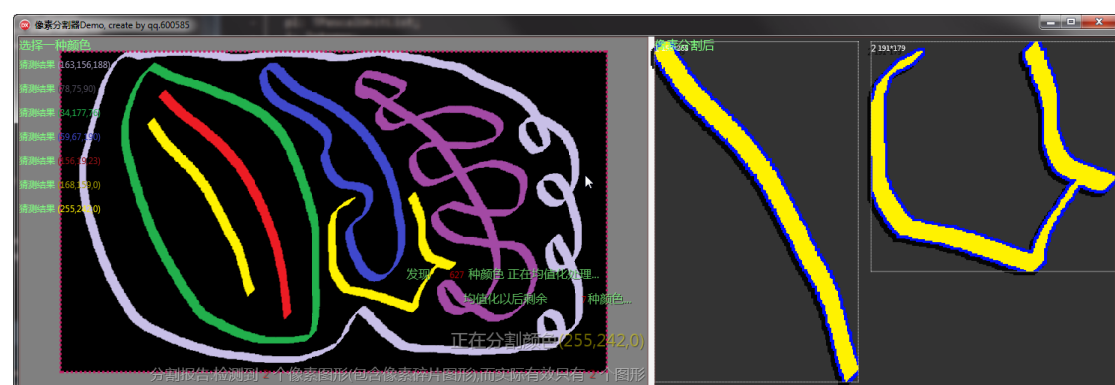
  const
    tab_coef_num: array[0..3, 0..16, 0..3] of vlc_bits_len =
    (
      (
        (1, 1), (0, 1), (0, 1), (0, 1),
        (5, 6), (1, 2), (0, 1), (0, 1),
        (7, 8), (4, 6), (1, 3), (0, 1),
        (7, 9), (6, 8), (5, 7), (3, 5),
        (7, 10), (6, 9), (5, 8), (3, 6),
        (7, 11), (6, 10), (5, 9), (4, 7),
        (15, 13), (6, 11), (5, 10), (4, 8),
        (11, 13), (14, 13), (5, 11), (4, 9),
      )
    );
```

新增使用文本解析引擎大规模排序 VCL 和 LCL 依赖库的 Demo

可以开发代码机器人，编程方式重构某些大型源码库

```
- |  
- procedure demo;  
480 var  
-   pl: TPascalUnitList;  
-   i: Integer;  
- begin  
-   // 使用这项技术制作的开源项目 https://github.com/PassByYou888/FFmpeg-Header  
-   pl := TPascalUnitList.Create;  
-  
-   // pl可以直接大规模加入代码文件，然后做依赖关系，做自动化的代码库分层  
-   // 甚至我们可以使用这项技术，来制作代码发行工具  
-  
490   pl.AddPascalCode(  
-     // 使用工具生成的代码  
-     'unit u1;'#13#10 +  
-     'interface'#13#10 +  
-     'uses u2,u3;'#13#10 +  
-     'implementation'#13#10 +  
-     '{$I test.inc}' + #13#10 +  
-     'end.'#13#10  
-   );  
500   pl.AddPascalCode(  
-     // 使用工具生成的代码  
-     'unit u2;'#13#10 +  
-     'interface'#13#10 +  
-     'uses u1;'#13#10 +  
-     'implementation'#13#10 +  
-     '{$I test.inc}' + #13#10 +  
-     'end.'#13#10  
-   );  
- end;  
- 
```

新增像素颜色的分割 Demo



内核 bug 修复

- 修复 1.19 的 OD 训练无故报告失败的问题

- 修复光栅尺度坐标系漏洞

假如图像如果是符合 2 次采样的 256×256 ，我们用浮点表示尺度是 1×1 ，在做采样换算时，使用 1×256 做地址换算，会导致采样成 $(1.0 + 1.0/256) \times 256$ 的地址，修复后的采样寻址公式为 $(1.0 \times (256 - 1))$ 。因为浮点坐标系没有 0 起始计数的概念。

下图是修复前后的软字体投影效果对比：注意边缘

