

NetworkTables API completa

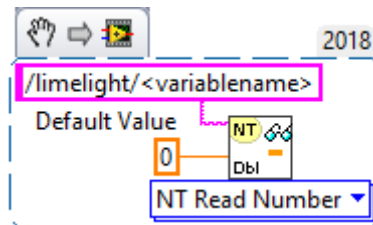
Dados básicos de segmento

Use os seguintes códigos:

Java

```
NetworkTableInstance.getDefault().getTable("limelight").getEntry("<variablename>").getDouble(0);
```

LABView



Para retornar os dados:

| Parâmetro | Descrição |
|-----------|---|
| tv | Se tem algum alvo válido (0 ou 1) |
| tx | Deslocamento horizontal do ponto alvo (LL1: -27° a 27° / LL2: -29,8° até 29,8°) |
| ty | Deslocamento vertical do ponto alvo (LL1: -20,5° a 20,5° / LL2: -24,85° a 24,85°) |
| ta | Área Alvo (0% de imagem a 100%) |
| tl | A latência da pipeline (ms). Adicione a <i>cl</i> para conseguir a latência total |
| cl | Captura a latência da pipeline (ms). Tempo entre o final da exposição da linha intermediária do sensor até o começo do pipeline de rastreamento |
| tshort | Comprimento lateral do lado mais curto da caixa de detecção (pixels) |
| tlong | Comprimento lateral do maior lado da caixa de detecção (pixels) |

| Parâmetro | Descrição |
|-----------|---|
| thor | Comprimento horizontal da caixa de detecção (0 - 320 pixels) |
| tvert | Comprimento vertical da caixa de detecção (0 - 320 pixels) |
| getpipe | Índice de pipeline ativo (0...9) |
| json | JSON completo dos alvos de segmentação |
| tclass | ID de classe do detector neural primário ou do classificador neural |
| tc | Obtenha a cor HSV média abaixo da região da mira como um <i>NumberArray</i> |

AprilTag e dados 3D

Use os seguintes códigos:

Java

```
NetworkTableInstance.getDefault().getTable("limelight").getEntry("<variablename>").getDoubleArray(new double[6]);
```

C++

```
nt::NetworkTableInstance::GetDefault().GetTable("limelight")->GetNumberArray("<variablename>",std::vector<double>(6));
```

Para retornar esse dado:

| Parâmetro | Descrição |
|------------------------|---|
| botpose | Transformação do robô no espaço do campo. Translação (X, Y, Z), Rotação (Roll, Pitch, Yaw), latência total (cl+tl) |
| botpose_wpiblue | Transformação do robô no espaço do campo (origem na <i>Driver Station</i> azul). Translação (X, Y, Z), Rotação (Roll, Pitch, Yaw), latência total (cl_tl) |
| botpose_wpired | Transformação do robô no espaço do campo (origem na <i>Driver Station</i> red). Translação (X, Y, Z), Rotação (Roll, Pitch, Yaw), latência total (cl_tl) |
| camerapose_targetspace | Transformação 3D da câmera no sistema de coordenadas da AprilTag em vista (array (6)) |
| targetpose_cameraspace | Transformação 3D da AprilTag em vista no sistema de coordenadas da câmera |

| Parâmetro | Descrição |
|-----------------------|---|
| targetpose_robotspace | Transformação 3D da AprilTag em vista no sistema de coordenadas do robô |
| camerapose_robotspace | Transformação 3D da câmera no sistema de coordenadas do robô |
| tid | ID da AprilTag em vista |

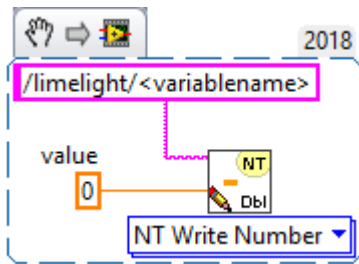
Controle de câmera

Use os seguintes códigos:

Java

```
NetworkTableInstance.getDefault().getTable("limelight").getEntry("<variablename>").setNumber(<value>);
```

LABView



C++

```
nt::NetworkTableInstance::GetDefault().GetTable("limelight")->PutNumber("<variablename>",<value>);
```

Python

```
NetworkTables.getTable("limelight").putNumber('<variablename>',<value>)
```

Para definir esse dado:

| ledMode | Definir o estado do LED |
|---------|-------------------------------------|
| [0] | Usa o modo do LED na pipeline atual |
| [1] | Desligado |
| [2] | Piscando |
| [3] | Ligado |

| camMode | Define o modo de operação |
|---------|--|
| 0 | Processador de visão |
| 1 | Câmera de <i>Driver</i> (Desabilita o processamento) |

| pipeline | Define a pipeline atual |
|----------|-------------------------|
| 0..9 | Define a pipeline 0...9 |

| stream | Define o modo de gravação da LimeLight |
|--------|--|
| 0 | Padrão - Transmissão simultânea se tiver uma webcam conectada oa robô |
| 1 | PiP Principal - A transmissão da câmera secundária é colocada no canto inferior direito ao da principal |
| 2 | PiP secundário - A transmissão da câmera principal é colocada no canto inferior direito ao da secundária |

| snapshot | Permite aos usuários tirar fotos durante a partida |
|----------|--|
| 0 | Reseta o modo de foto |
| 1 | Tira uma foto |

| crop | (Matriz) Define o recorte retângular. A pipeline deve utilizar o corte padrão da interface <i>web</i> . A matriz deve ter 4 entradas. |
|------|---|
| [0] | X0 - Mín ou Máx valor de X do recorte retângular (-1 a 1) |
| [1] | X1 - Mín ou Máx valor de X do recorte retângular (-1 a 1) |
| [2] | Y0 - Mín ou Máx valor de Y do recorte retângular (-1 a 1) |
| [3] | Y1 - Mín ou Máx valor de Y do recorte retângular (-1 a 1) |

camerapose_robotspace_set | (Matriz) Define a posição da câmera no sistema de coordenadas do robô

Java

```
double[] cropValues = new double[4];
cropValues[0] = -1.0;
cropValues[1] = 1.0;
cropValues[2] = -1.0;
cropValues[3] = 1.0;

NetworkTableInstance.getDefault().getTable("limelight").getEntry("crop").setDoubleArray(cropValues);
```

C++

Python

Os algoritmos em Python permitem dados arbitrários de entrada e saída.

lpython - NumberArray enviado pelo *script* de python. Esse dado é acessível pelo código do robô.

lrobot - NumberArray enviado pelo robô. É acessível pelo algoritmo de python.

Contornos

Cantos:

Habilite *send contours* na aba de *Output* para transmitir as coordenadas dos cantos

tcornxy - Matriz das coordenadas [x0, y0, x1, y1...]

Alvos brutos:

O Limelight envia três contornos brutos para a NetworkTables que não são influenciados pelo modo de agrupamento. Ou seja, eles são filtrados com os parâmetros da sua tubulação, mas nunca agrupados. X e Y são retornados no espaço de tela normalizado (-1 a 1) em vez de graus.

| Dados | Descrição |
|----------------------|-----------------------------------|
| tx0 | Espaço de tela X |
| ty0 | Espaço de tela Y |
| ta0 | Área (0% de imagem a 100%) |
| ts0 | Inclinação ou rotação (-90° a 0°) |
| tx1 | Espaço de tela X |
| ty1 | Espaço de tela Y |
| ta1 | Área (0% de imagem a 100%) |
| ts1 | Inclinação ou rotação (-90° a 0°) |
| tx2 Espaço de tela X | |
| ty2 | Espaço de tela Y |
| ta2 | Área (0% de imagem a 100%) |
| ts2 | Inclinação ou rotação (-90° a 0°) |

Mira bruta:

Se estiver usando dados de direcionamento brutos, ainda é possível utilizar suas miras calibradas:

| Dados | Descrição |
|-------|----------------------------|
| cx0 | Mira A X no espaço de tela |
| cy0 | Mira A Y no espaço de tela |
| cx1 | Mira B X no espaço de tela |
| cy1 | Mira B Y no espaço de tela |

Revisão #3
Criado 22 janeiro 2024 13:52:21 por Enzo Coutinho
Atualizado 23 janeiro 2024 11:52:32 por Enzo Coutinho