

Limelight

Limelight é câmera inteligente plug-and-play desenvolvida especificamente para a FIRST Robotics Competition. A utilização da Limelight é fácil o suficiente para equipes sem experiência em visão ou mentores especializados e poderosa o suficiente para equipes experientes que precisam de uma solução de visão confiável e pronta para a competição.

- [Introdução](#)
 - [Limelight - Visão ao seu robô sem código](#)
 - [Montagem](#)
 - [Conexão](#)
 - [Atualizando o LimelightOS](#)
 - [Setup de Comunicação/Network](#)
 - [Estado das luzes](#)
 - [Início a Programação](#)
 - [Melhores práticas](#)
 - [Calibração de mira](#)
 - [Configuração Pipeline](#)
- [Pipeline de AprilTags](#)
 - [Rastreando AprilTags](#)
 - [AprilTags 3D](#)
 - [Localização do robô com MegaTags](#)
 - [Sistema de coordenadas 3D em detalhes](#)
 - [Ferramenta de construção de mapa](#)
- [Pipelines de Redes Neurais](#)
 - [Começando com Redes Neurais](#)
 - [Treinando um modelo de classificação customizado](#)
 - [Treinando um modelo de detecção customizado](#)
- [Pipeline com Python](#)

- [Pipelines com Python](#)
- [Pro Pipelines com Python](#)
- [API's](#)
 - [NetworkTables API completa](#)
- [Aviso](#)

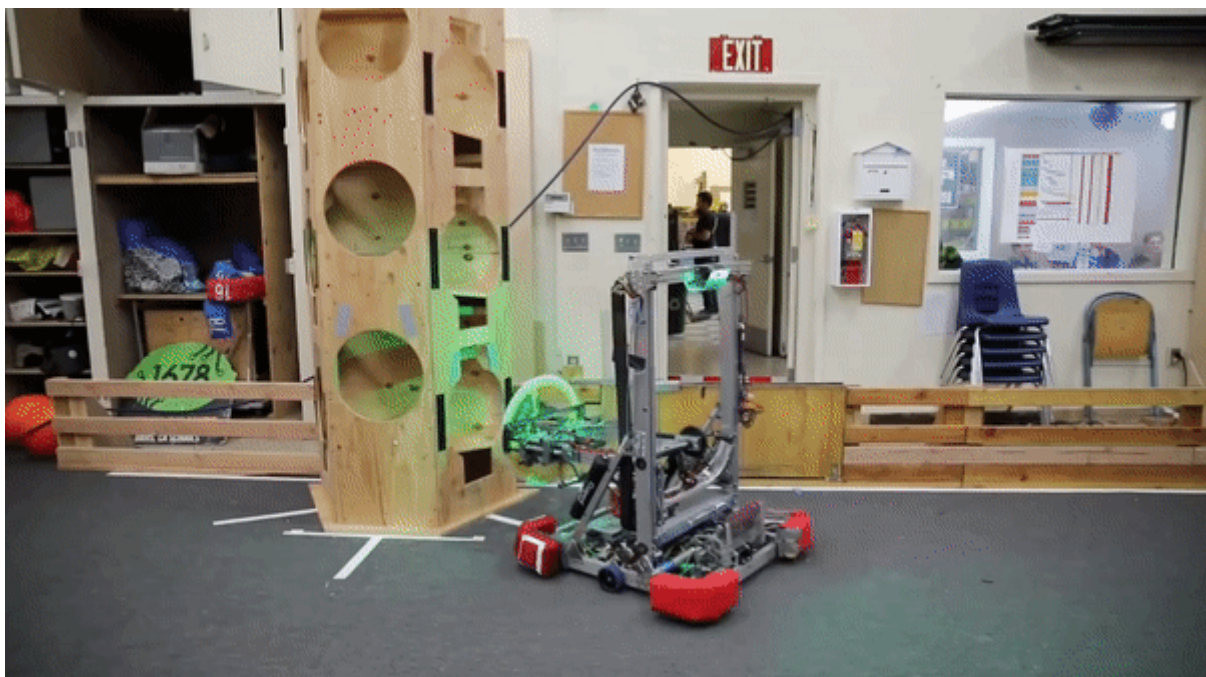
Introdução

Limelight - Visão ao seu robô sem código

A Limelight foi projetada para tornar a percepção de um robô tão fácil e confiável possível sem sacrificar nenhum desempenho!

A Limelight é fácil o suficiente para iniciantes completos e poderoso o suficiente para profissionais.

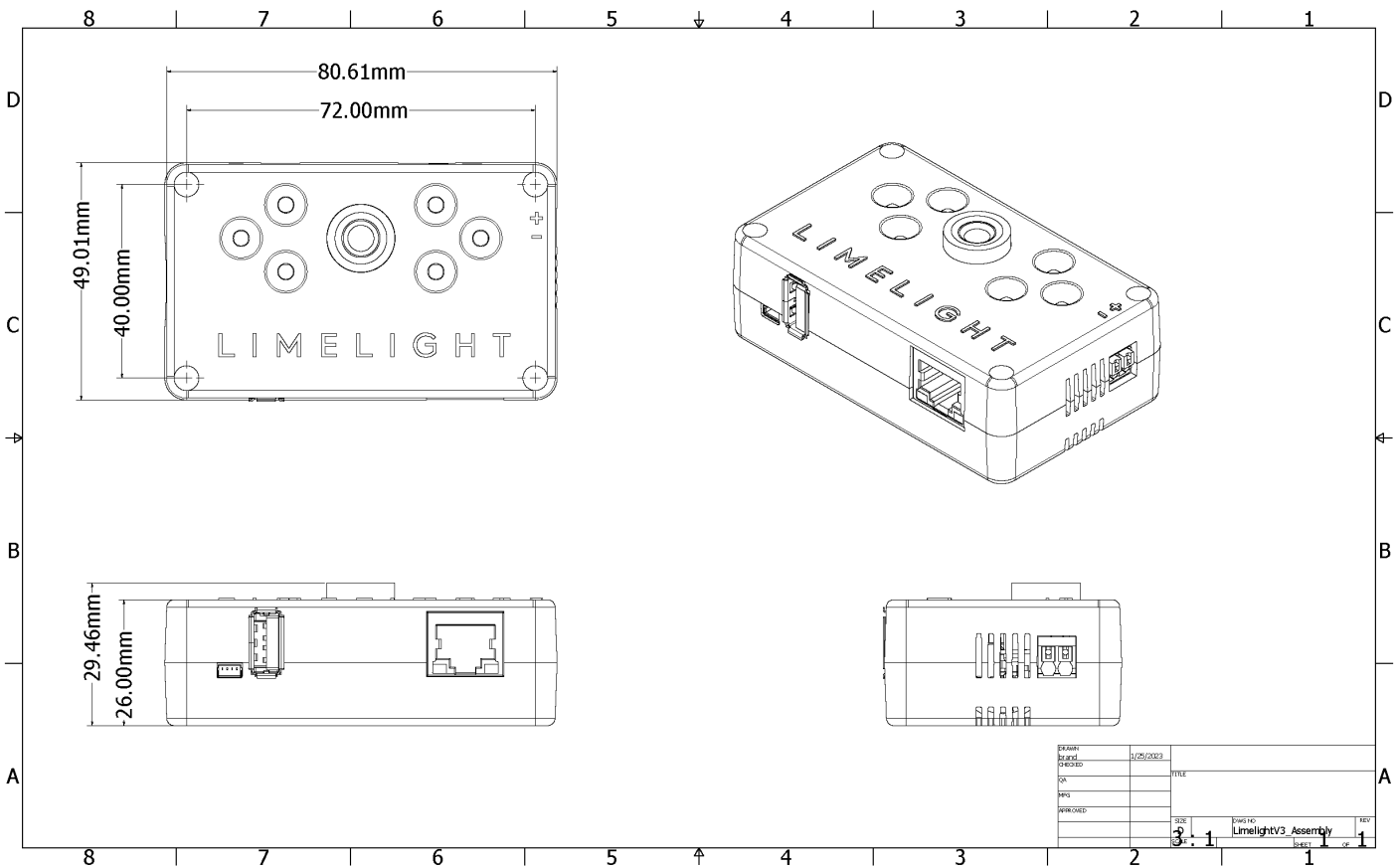
- Configure pipelines de visão computacional sem código para bolhas de cor, AprilTags, redes neurais e muito mais usando a interface web integrada.
- Escreva pipelines personalizadas em Python com SnapScript usando tensorflow, opencv e muito mais usando a interface web integrada ou o Visual Studio Code.
- O hardware Limelight integra um sensor de imagem MIPI-CSI de alta largura de banda e baixa latência, um computador arm64, condicionamento de energia e o LimelightOS.
- O LimelightOS suporta os protocolos REST/HTTP, Websocket, Modbus e NetworkTables, além de formatos de saída JSON, Protobuf e bruto.



Montagem

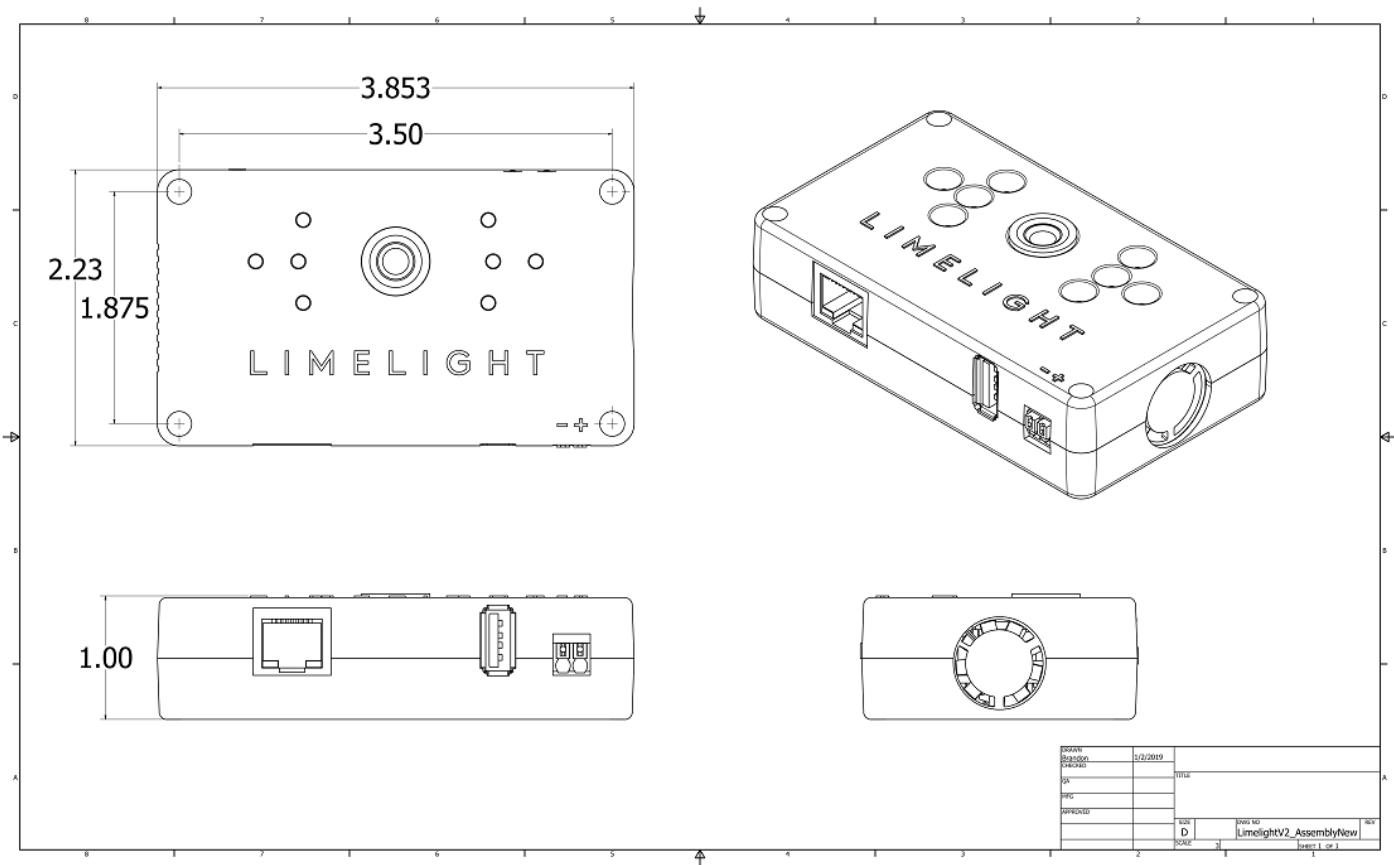
Limelight 3

Use quatro parafusos de 1 1/2" #10-32 ou #10-24 e porcas de nylon para prender sua Limelight.



Limelight 2/2+

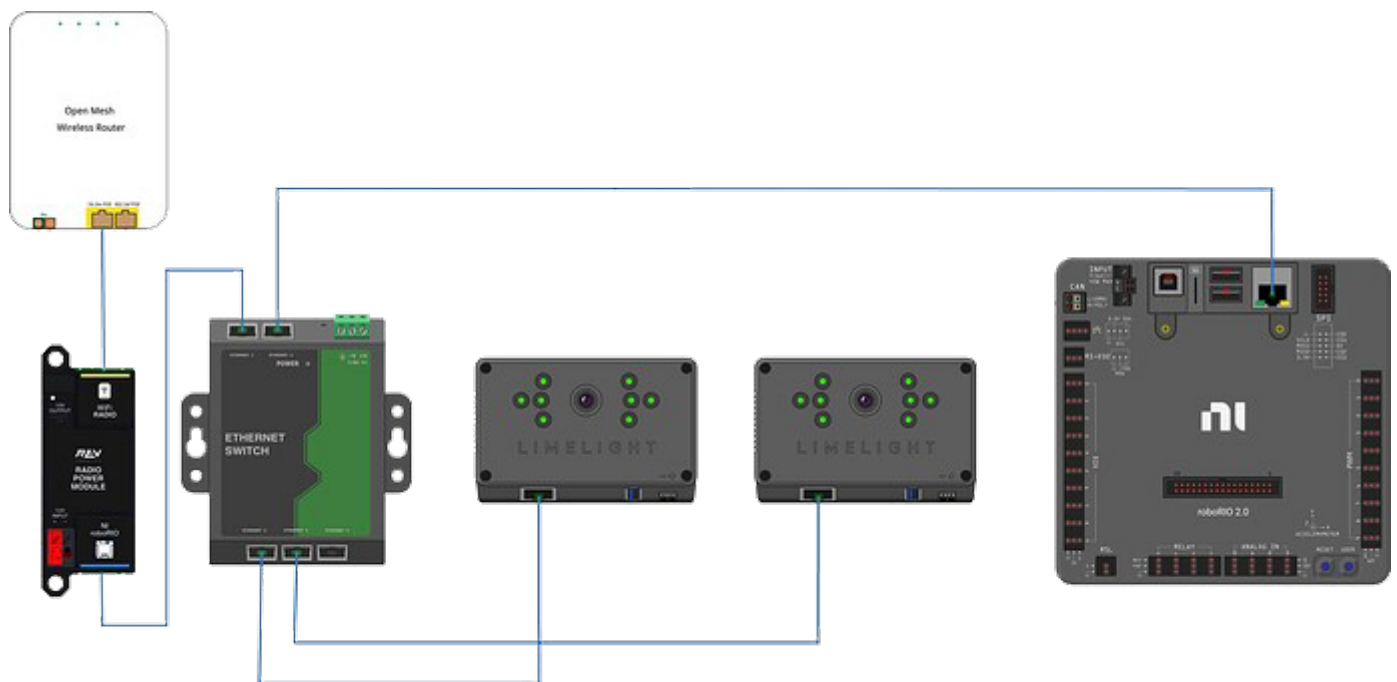
Use quatro parafusos de 1 1/2" #10-32 ou #10-24 e porcas de nylon para prender sua Limelight.



Conexão

A Limelight recebe 12V de alimentação, porém é construída para funcionar até 4.5V. Os LEDs tem brilho constante até 7V

Não utilize o Radio Power Module da REV para alimentar sua Limelight. A tensão é muito alta



Credito - Christian Femia

Conexão Padrão

- Não conecte os cabos na VRM
- Passe dois fios do seu Limelight para um slot no seu PDP/PDH (NÃO no seu VRM).
- Adicione qualquer disjuntor (5A, 10A, 20A, etc.) ao mesmo slot no seu PDP.
- Passe um cabo ethernet do seu Limelight para o rádio.

Conexão POE

PoE permite que você adicione tanto energia quanto a conexão de rede a sua Limelight por meio de um cabo Ethernet.

Isto não é padrão IEEE 802.3 (44V-48V) PoE - é por isso que você deve usar um injetor passivo com 12V.

- Conecte um Injetor PoE Passivo ao seu PDP (NÃO ao seu VRM).
- Adicione qualquer disjuntor (5A, 10A, 20A, etc.) ao mesmo slot no seu PDP.
- Passe um cabo ethernet do seu Limelight para o seu injetor POE passivo.

Melhores Práticas de Fiação

- Adicione alívio de tensão a todos os cabos de energia e ethernet que vão para o sua LL.
- Cole com cola quente todas as conexões.
- Adicione um switch de rede ao seu robô para permitir tethering ethernet durante os eventos e para evitar a segunda porta de rádio.
- Use cabos Cat6 trançados com fios flexíveis.
- Não é recomendado o uso da segunda porta de rádio. Roteie todos os dispositivos através do seu switch de rede, se possível.

Atualizando o LimelightOS

Siga esse guia para atualizar sua Limelight para a versão mais recente do LimelightOS

Limelight 3

- Desligue a sua Limelight.
- Baixe os últimos drivers USB, a imagem do sistema operacional Limelight e a ferramenta Balena Flash da [Página de Downloads da Limelight](#).
- Conecte um cabo USB->USB-C do seu laptop ao seu Limelight. Seu Limelight ligará automaticamente.
- Execute o "Balena Etcher" como administrador.
- Pode levar até 20 segundos para que seu computador reconheça a câmera.
- Selecione a última imagem .zip na sua pasta de downloads.
- Selecione um dispositivo "Módulo de Computação" no menu "Drives".
- Clique em "Flash".
- Uma vez concluída a gravação, remova o cabo USB do seu Limelight.

Limelight 2/2+

- Desligue o seu Limelight.
- Baixe os mais recentes drivers USB, a imagem do sistema operacional Limelight e a ferramenta Balena Flash da [Página de Downloads](#).
- Conecte um cabo USB->MicroUSB do seu laptop para o seu Limelight. O Limelight ligará automaticamente.
- Execute o "Balena Etcher" como administrador.
- Pode levar até 20 segundos para que o seu computador reconheça a câmera.
- Selecione a imagem .zip mais recente na sua pasta de downloads.
- Selecione um dispositivo "Compute Module" no menu "Drives".
- Clique em "Flash".
- Uma vez que a gravação esteja completa, remova o cabo USB do seu Limelight.

Conecte o cabo USB-C apenas durante a gravação. O Limelight entra em um modo especial de flash enquanto o cabo microUSB está conectado. Você não poderá acessar a interface web enquanto o Limelight estiver no modo flash.

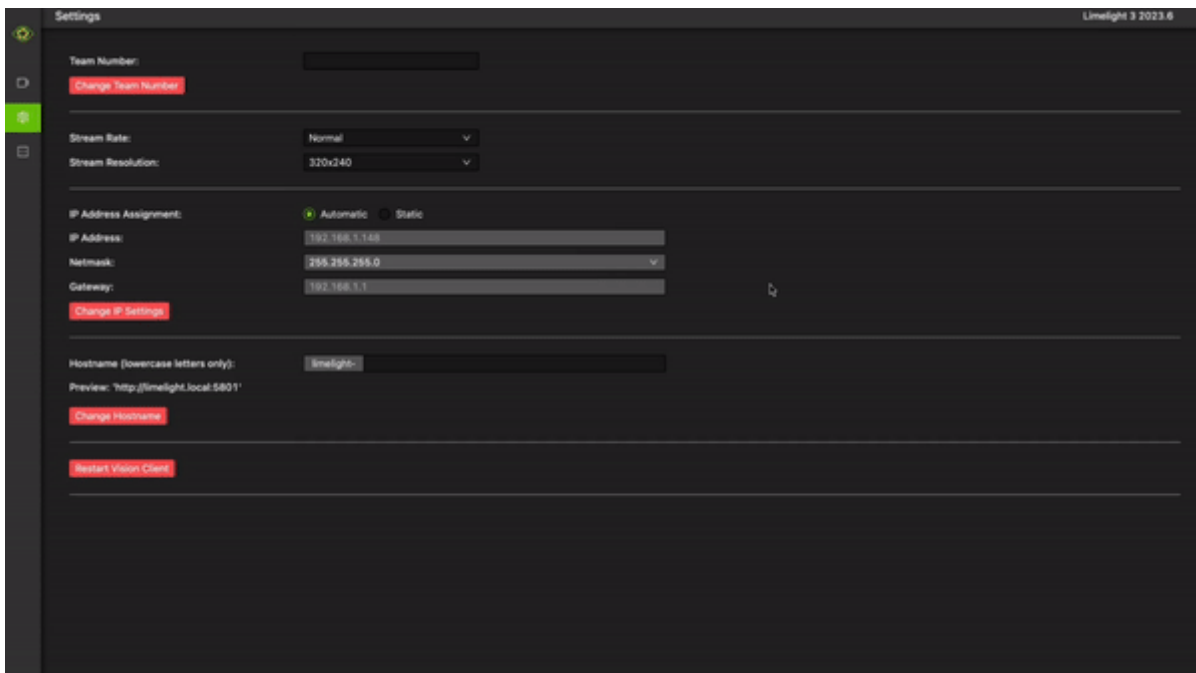
Setup de Comunicação/Network

Nós recomendamos fortemente seguir as instruções de IP estático para confiabilidade durante eventos. Siga estes passos antes de começar:

- Vá em adicionar/remover programas no Windows e procure por "bonjour"
- Quantos itens você vê?
 - Se houver dois (2) itens com "bonjour" em seus nomes, desinstale "bonjour print services"
 - Se não houver (0) itens com "bonjour" em seus nomes, instale bonjour da nossa página de Downloads.
- Reinicie seu Limelight/Robô e computador.
- Baixe a Ferramenta de Localização do Limelight da página de [Downloads](#)

Definir Número da Equipe

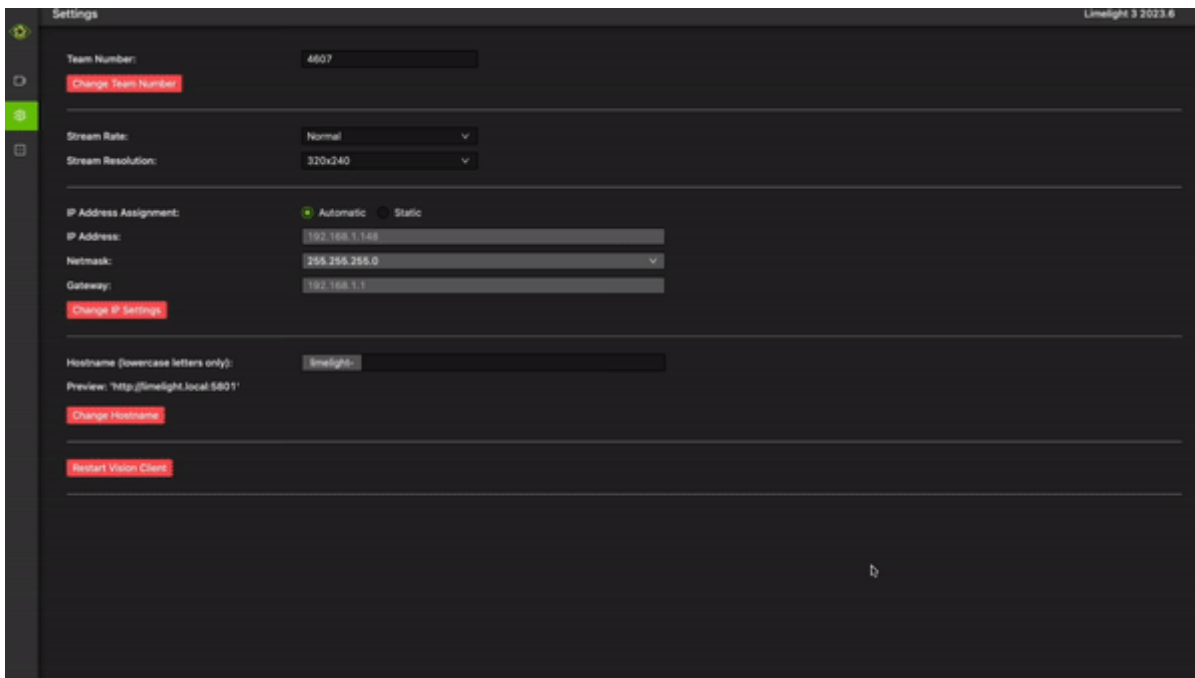
- Ligue o seu robô e conecte seu computador à rede do seu robô.
- Após o seu Limelight piscar seus LEDs verdes, abra a Ferramenta de Localização do Limelight e procure pelo seu Limelight ou navegue para <http://limelight.local:5801>. Este é o painel de configuração.
- Navegue até a aba "Configurações" no lado esquerdo da interface.
- Insira o número da sua equipe e pressione o botão "Atualizar Número da Equipe".



Definir Endereço IP

Endereçamento Estático

- Mude o seu "Atribuição de IP" para "Estático".
- Defina o endereço IP do seu Limelight para "10.TE.AM.11".
- NOTA: Equipes com zeros precisam prestar atenção especial:
 - A equipe 916 usa 10.9.16.xx,
 - A equipe 9106 usa 10.91.6.xx
 - A equipe 9016 usa 10.90.16.xx
- Defina a Máscara da Sub-rede para "255.255.255.0".
- Defina o Gateway para "10.TE.AM.1".
- Clique no botão "Atualizar".
- Dê ao seu roboRIO o seguinte endereço IP estático: "10.TE.AM.2"
- Recicle a energia do seu robô.
- Agora você poderá acessar o seu painel de configuração em 10.TE.AM.11:5801, e a transmissão da sua câmera em 10.TE.AM.11:5800.



Endereçamento Dinâmico

- Mude sua "Atribuição de IP" para "Automático".
- Clique no botão "Atualizar".
- Reinicie o Robô.
- Você pode continuar acessando seu painel de configuração em <http://limelight.local:5801>, e sua transmissão de câmera em <http://limelight.local:5800>

Conecte o cabo USB-C apenas durante a gravação de imagem. O Limelight entra em um modo especial de flash enquanto o cabo microUSB está conectado. Você não será capaz de acessar a interface web enquanto o Limelight estiver em modo flash.

Por que recomendamos o uso de um endereço IP estático?

Primeiro, isso reduz vários segundos do tempo de inicialização do Limelight. Segundo, as equipes historicamente tiveram problemas em campos FRC reais e com firmwares de rádio de eventos.

Recomendamos configurar endereços IP estáticos no seu robo-rio e driverstation também. As configurações de rede para usar em todos esses dispositivos podem ser encontradas perto da metade inferior desta [página web](#).

Como eu reseto o endereço IP?

Após o seu Limelight iniciar, segure o botão de configuração por 10 segundos. Na próxima vez

que o Limelight iniciar, sua configuração de rede será redefinida para endereçamento dinâmico.

Antes de se Conectar ao Campo

- Configure um IP estático para o seu laptop.
 - IP: 10.TE.AM.5
 - Máscara de Sub-rede: 255.0.0.0
 - Gateway: 10.TE.AM.1
- Configure um IP estático para o seu RIO.
 - IP: "10.TE.AM.2"
 - Máscara de Sub-rede: 255.255.255.0 <- NOTE A DIFERENÇA AQUI
 - Gateway: 10.TE.AM.1
- Dê as suas Limelights nomes de host únicos (se estiver usando vários).
- Dê as suas Limelights configurações de IP estático únicas.
 - Comece sempre com endereços ".11" e vá subindo. (10.9.87.11, etc.)
 - O uso de outros endereços pode causar mau funcionamento das suas unidades quando conectado ao FMS.
 - IP: "10.TE.AM.11"
 - Máscara de Sub-rede: 255.255.255.0
 - Gateway: 10.TE.AM.1

Nota: Equipes com zeros precisam prestar atenção especial:

- A equipe 916 utiliza 10.9.16.xx
- A equipe 9106 utiliza 10.91.6.xx
- A equipe 9016 utiliza 10.90.16.xx

Estado das luzes

Luz de status verde

A luz de status verde piscará lentamente se nenhum alvo for detectado pela pipeline atual. Ela piscará rapidamente se algum alvo for detectado pela pipeline atual.

Luz de status amarela

A luz de status amarela piscará quando um endereço IP estático não tiver sido atribuído. Se um endereço IP estático for atribuído, a luz permanecerá constantemente acesa ou apagada, sem piscar.

LEDs de iluminação verde

Os LEDs de iluminação verde no Limelight 1, 2 e 3 são controláveis via interface web e várias APIs, mas existem alguns padrões especiais de piscar projetados para ajudar a solucionar problemas de hardware e software:

- Piscar alternado Esquerda/Direita ou Superior/Inferior: O cabo interno da câmera se soltou ou o sensor de imagem sofreu danos.
- Piscar Rápido (todos os LEDs): O botão de reinicialização de rede foi mantido pressionado por pelo menos 10 segundos.
- Sequência de Inicialização Repetida (três piscadas ou várias piscadas de desvanecimento): O software está falhando, possivelmente devido a danos no hardware.

Início a Programação

O Limelight suporta os protocolos REST/HTTP, Websocket, Modbus e NetworkTables para dados de mira, dados de status e configuração ao vivo. Formatos de saída JSON, Protobuf e bruto estão disponíveis. Consulte a seção de APIs da documentação para obter mais informações.

Para equipes de FRC, o protocolo recomendado é o [NetworkTables](#). O Limelight envia todos os dados de mira, incluindo um despejo completo em JSON, para o NetworkTables a 100hz. As equipes também podem definir controles, como ledMode, janela de corte e mais via NetworkTables. As equipes de FRC podem usar as bibliotecas Limelight Lib Java e C++ para começar com o Limelight em segundos. Limelight Lib é a maneira mais fácil de começar.

Biblioteca Limelight:

[Java](#), [C++](#)

Java

```
double tx = LimelightHelpers.getTX("");
```

C++

```
#include "LimelightHelpers.h"
```

```
double tx = LimelightHelpers::getTX("");  
double ty = LimelightHelpers::getTY("");
```

Python

```
wip
```

NetworkTables

Java

```
import edu.wpi.first.wpilibj.smartdashboard.SmartDashboard;
import edu.wpi.first.networktables.NetworkTable;
import edu.wpi.first.networktables.NetworkTableEntry;
import edu.wpi.first.networktables.NetworkTableInstance;
```

```
NetworkTable table = NetworkTableInstance.getDefault().getTable("limelight");
NetworkTableEntry tx = table.getEntry("tx");
NetworkTableEntry ty = table.getEntry("ty");
NetworkTableEntry ta = table.getEntry("ta");

//Lê os valores periodicamente
double x = tx.getDouble(0.0);
double y = ty.getDouble(0.0);
double area = ta.getDouble(0.0);

//Manda o valor para a smart dashboard periodicamente
SmartDashboard.putNumber("LimelightX", x);
SmartDashboard.putNumber("LimelightY", y);
SmartDashboard.putNumber("LimelightArea", area);
```

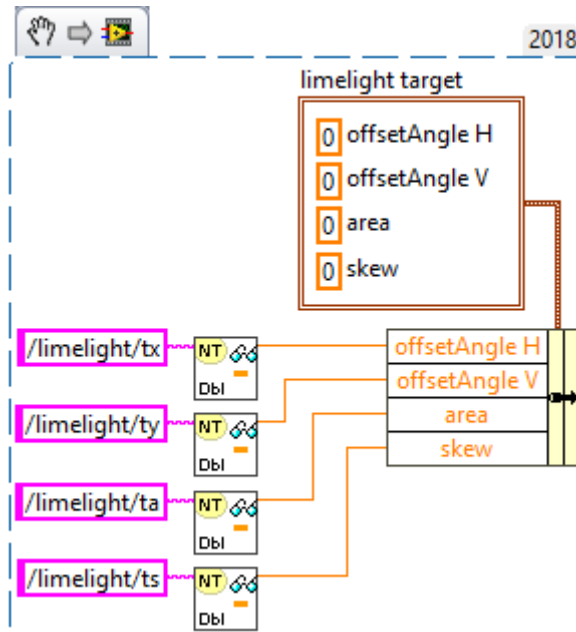
C++

```
#include "frc/smartdashboard/Smartdashboard.h"
#include "networktables/NetworkTable.h"
#include "networktables/NetworkTableInstance.h"
#include "networktables/NetworkTableEntry.h"
#include "networktables/NetworkTableValue.h"
#include "wpi/span.h"
```

```
std::shared_ptr<nt::NetworkTable> table = nt::NetworkTableInstance::GetDefault().GetTable("limelight");
double targetOffsetAngle_Horizontal = table->GetNumber("tx", 0.0);
double targetOffsetAngle_Vertical = table->GetNumber("ty", 0.0);
```

```
double targetArea = table->GetNumber("ta",0.0);  
double targetSkew = table->GetNumber("ts",0.0);
```

LabVIEW



Python

```
import cv2  
import numpy as np  
  
# runPipeline() é chamado todo frame pelo backend da Limelight  
def runPipeline(image, llobot):  
    # converte a imagem de input para o espaço de cor do HSV  
    img_hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)  
    # converte o hsv para uma imagem binaria removendo qualquer pixel  
    # que não sejam de acordo com os seguintes valores minimos e máximos do HSV  
    img_threshold = cv2.inRange(img_hsv, (60, 70, 70), (85, 255, 255))  
  
    # ache contornos na nova imagem binaria  
    contours, _ = cv2.findContours(img_threshold,  
    cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)  
  
    largestContour = np.array([[]])  
  
    # inicializa um array vazio para mandar devolta ao robô
```

```
lpython = [0,0,0,0,0,0,0]
```

```
# se os contornos foram detectados, desenhe eles
```

```
if len(contours) > 0:
```

```
    cv2.drawContours(image, contours, -1, 255, 2)
```

```
    # grava o maior contorno
```

```
    largestContour = max(contours, key=cv2.contourArea)
```

```
    # pega a caixa delimitadora não rotacionada que envolve o contorno
```

```
    x,y,w,h = cv2.boundingRect(largestContour)
```

```
    # desenha a caixa delimitadora não rotacionada
```

```
    cv2.rectangle(image,(x,y),(x+w,y+h),(0,255,255),2)
```

```
    # grave alguns dados customizados para mandar devolta ao robô
```

```
    lpython = [1,x,y,w,h,9,8,7]
```

```
#retorna o maior contor para a mira da LL, a imagem modificada, e os dados customizados
```

```
return largestContour, image, lpython
```

Melhores práticas

Lista de Verificação de Preparação para o Evento

- Baixe e faça backup de todas as pipelines para o seu laptop de programação.
- Faça o download de uma cópia da imagem mais recente do Limelight para o seu laptop de programação.
- Registre uma lista das suas pipelines e seus índices.
 - 1 - Dual Target Low
 - 2 - Dual Target High Cargo
- Adicione alívios de tensão a todos os cabos de energia e Ethernet que vão para o seu Limelight.
- Considere aplicar cola quente a todas as conexões.
- Certifique-se de estar usando um painel de controle (Smartdashboard, Shuffleboard) e não um navegador da web para visualizar o stream durante treinos e eventos. Os controles web padrão não têm a capacidade de reconectar automaticamente a streams após desconexões acidentais, mas o SmartDashboard e o Shuffleboard têm reconexão automática incorporada por padrão.
- Adicione um switch de rede ao seu robô para habilitar a conexão Ethernet enquanto estiver em um evento e evitar a segunda porta de rádio.
- Use cabos Cat6 torcidos com fios flexíveis. Não use cabos Ethernet planos.
- Não use a segunda porta de rádio. Encaminhe todos os dispositivos por meio do seu switch de rede, se possível.
- Configure o encaminhamento de porta (Port Forwarding) para permitir a comunicação do Limelight ao ser conectado ao seu robô via USB.
 - Encaminhe as portas 5800, 5801, 5802, 5803, 5804, 5805, 5806 e 5807.

Java

```
import edu.wpi.first.wpilib.net.PortForwarder;

@Override
public void robotInit()
{
    // Certifique-se de configurar o encaminhamento de porta apenas uma vez no código do robô.
    // Não coloque essas chamadas de funções em funções periódicas
```

```
for (int port = 5800; port <= 5807; port++) {  
    PortForwarder.add(port, "limelight.local", port);  
}  
}
```

C++

```
import edu.wpi.first.wpiutil.net.PortForwarder;  
  
@Override  
public void robotInit()  
{  
    // Certifique-se de configurar o encaminhamento de porta apenas uma vez no código do robô.  
    // Não coloque essas chamadas de funções em funções periódicas  
    for (int port = 5800; port <= 5807; port++) {  
        PortForwarder.add(port, "limelight.local", port);  
    }  
}
```

Calibração durante o evento

- Role seu robô até cada alvo no campo.
 - Certifique-se de que a segmentação está funcionando corretamente. Mude para a visualização "threshold" durante este processo (localizada abaixo do fluxo de imagem).
 - Role seu robô próximo ao alvo e longe do alvo. Garanta que as miras estejam calibradas corretamente.
 - Enquanto estiver longe do alvo, gire seu robô para a esquerda e para a direita ~ 30 graus para garantir que outros alvos não sejam rastreados erroneamente.
 - Consulte a seção de ajuste abaixo para mais dicas de ajuste.
 - Certifique-se de que nenhum outro elemento do campo/arena esteja sendo rastreado acidentalmente. Verifique seus filtros de área e proporção se estiver captando as luzes da arena.
 - Tire capturas de tela de todos os alvos e rastreamentos incorretos. Você pode usá-los para ajustar suas pipelines nos boxes.

Melhorando o pipeline

- Utilize a exposição mais baixa possível e aumente o deslocamento do nível de preto até que as luzes do campo e reflexos de LEDs sejam removidos da imagem.

- Teste sua segmentação enquanto estiver longe e inclinado para longe do seu alvo.
- Use o recurso "Smart Speckle Rejection" da versão 2019.7 para filtrar reflexos indesejados de LEDs.

Antes de conectar ao campo

- Configure seu laptop com uma configuração de IP estático.
 - IP: 10.TE.AM.5
 - Máscara de Sub-rede: 255.0.0.0
 - Gateway: 10.TE.AM.1
- Configure seu RIO com uma configuração de IP estático.
 - IP: "10.TE.AM.2"
 - Máscara de Sub-rede: 255.255.255.0 <- ATENÇÃO PARA A DIFERENÇA AQUI
 - Gateway: 10.TE.AM.1
- Dê nomes de host exclusivos para seus Limelights (se estiver usando vários).
- Forneça configurações de IP estático únicas para seus Limelights.
 - Comece sempre com endereços terminados em ".11" e vá aumentando. (10.9.87.11, etc.)
 - O uso de outros endereços pode fazer com que suas unidades apresentem mau funcionamento quando conectadas ao FMS.
 - IP: "10.TE.AM.11"
 - Máscara de sub-rede: 255.255.255.0
 - Gateway: "10.TE.AM.1" Nota: As equipes com zeros precisam prestar atenção especial:

```
- _Team 916 uses 10.9.16.xx,_  
- _Team 9106 uses 10.91.6.xx_  
- _Team 9016 uses 10.90.16.xx_
```

Informação adicional: <https://docs.wpilib.org/en/stable/docs/networking/networking-introduction/ip-configurations.html>

Lista de checagem Pré-Partida

- Verifique todos os cabos de energia e Ethernet que vão para seus Limelights.
- Verifique todas as conexões elétricas quanto a folgas e fios desgastados.
- Verifique todos os parafusos de fixação / abraçadeiras / fita adesiva.
- Observe precauções contra descarga eletrostática (ESD) o tempo todo.

Largura de banda

- Algumas equipes utilizam dois Limelights com duas câmeras USB, mantendo-se bem abaixo do limite de largura de banda. Siga as etapas abaixo para reduzir a largura de banda.
- Em vez de usar o modo de condução, crie uma pipeline "driver". Reduza a exposição para diminuir a largura de banda do stream.
- Está usando uma câmera USB? Utilize a chave NT "stream" para ativar o modo picture-in-picture. Isso reduzirá drasticamente a largura de banda do stream.
- Configure a taxa de stream para "baixa" na página de configurações se o streaming não for crítico para a condução.
- Utilize a opção de stream 160x120 introduzida na versão 2019.7.

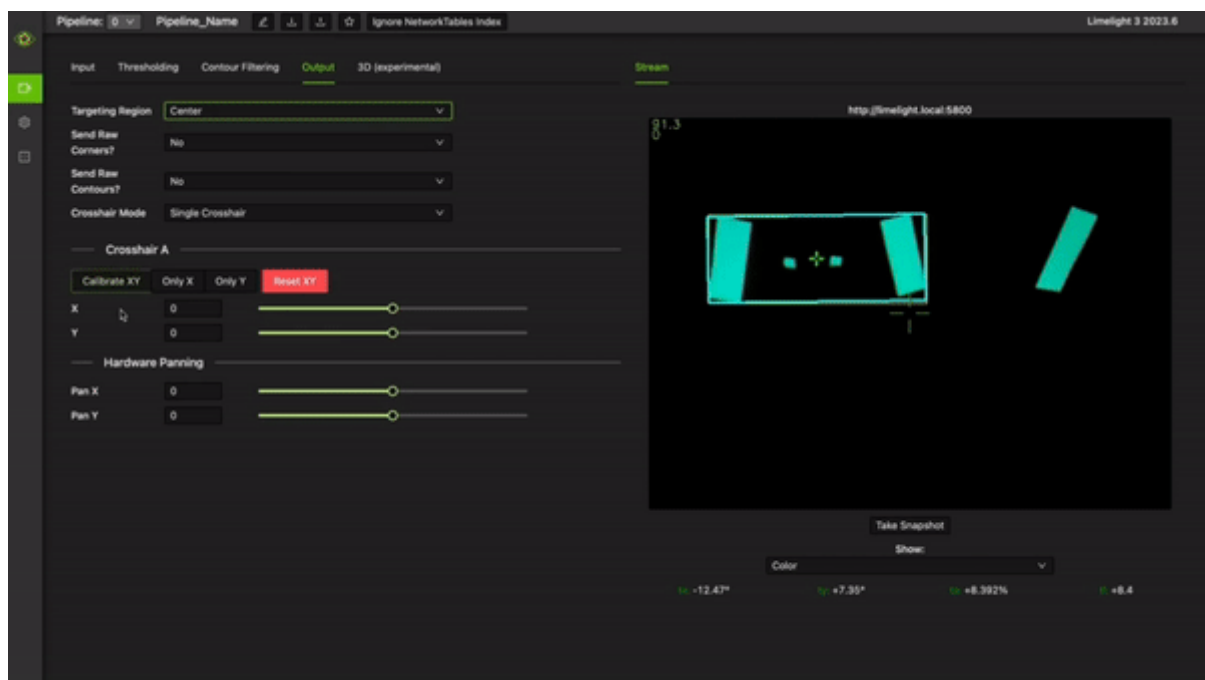
Solução de erros

- Tente acessar o stream em IP:5800 com um navegador da web. Isso deve ajudar a determinar a origem dos seus problemas.
- Reinicie o seu painel de controle.
- Reinicie o seu computador.
- Reinicie o seu robô se o campo tiver sido redefinido.
- Cabos Ethernet danificados podem ser a causa de problemas intermitentes de rede.
- Sempre use configurações de IP estático no campo.

Calibração de mira

As miras do Limelight transformam a calibração angular 2D em um processo simplificado. Em vez de armazenar desvios angulares no código, você pode alinhar manualmente seu robô a um alvo e clicar no botão "calibrar".

Calibrar uma mira move o "zero" dos seus dados de mira. Isso é muito útil se o Limelight não estiver perfeitamente centrado no seu robô.



Modo de mira única

Posicione o seu robô na sua localização e rotação ideais para pontuar e clique em "calibrar". Agora, um tx e ty de "zero" equivalem a um robô perfeitamente alinhado. Se o seu robô precisar ser recalibrado para um novo campo, simplesmente realize uma partida de treino para encontrar o alinhamento perfeito para o seu robô e clique em "calibrar" durante a partida.

Modo de mira dupla

Imagine um robô com uma câmera ou atirador fora do eixo que precisa lançar objetos do jogo em um gol de várias posições no campo. À medida que o robô se aproxima do gol, sua mira deve ajustar em tempo real para compensar. O modo de mira dupla foi criado para essa funcionalidade.

Alinhe o seu robô na sua posição e rotação mais próxima de pontuação e calibre a mira "A". Alinhe o seu robô na sua posição e rotação mais distante de pontuação e calibre a mira "B". Ao calibrar no modo de mira dupla, as miras também armazenam um valor de área. Você perceberá que, à medida que o seu robô se move entre as distâncias mínima e máxima de pontuação, a mira se move entre a mira "A" e a mira "B". Isso é feito verificando a área do alvo e comparando-a com as duas áreas de alvo vistas durante a calibração.

Configuração Pipeline

O LimelightOS armazena até 10 pipelines de visão exclusivas. Você pode alterar a pipeline ativa em tempo real com o LimelightLib.

Para editar várias pipelines, você deve primeiro marcar a caixa "Ignore NetworkTables Index" na interface web. Isso forçará o robô a permitir temporariamente que você altere o índice da pipeline por meio da interface web, em vez de por meio do NetworkTables.

Para baixar suas pipelines para backup e compartilhamento, basta clicar no botão "download" próximo ao nome da sua pipeline. Para carregar uma pipeline, clique no botão "upload".

Aqui está um exemplo de um robô que utiliza duas pipelines:

Exemplo de Robô com Duas Pipelines

A primeira pipeline é ajustada para identificar faixas verticais únicas. A segunda pipeline é ajustada para encontrar uma combinação de duas faixas horizontais. O código para este robô está disponível no estudo de caso "Aim and Range".

Observe que, quando o robô alterna entre pipelines, a interface web carrega automaticamente a nova pipeline.

Aba de entrada

A guia de entrada (Input Tab) oferece controles para alterar a imagem da câmera antes que ela seja processada pela pipeline de processamento.

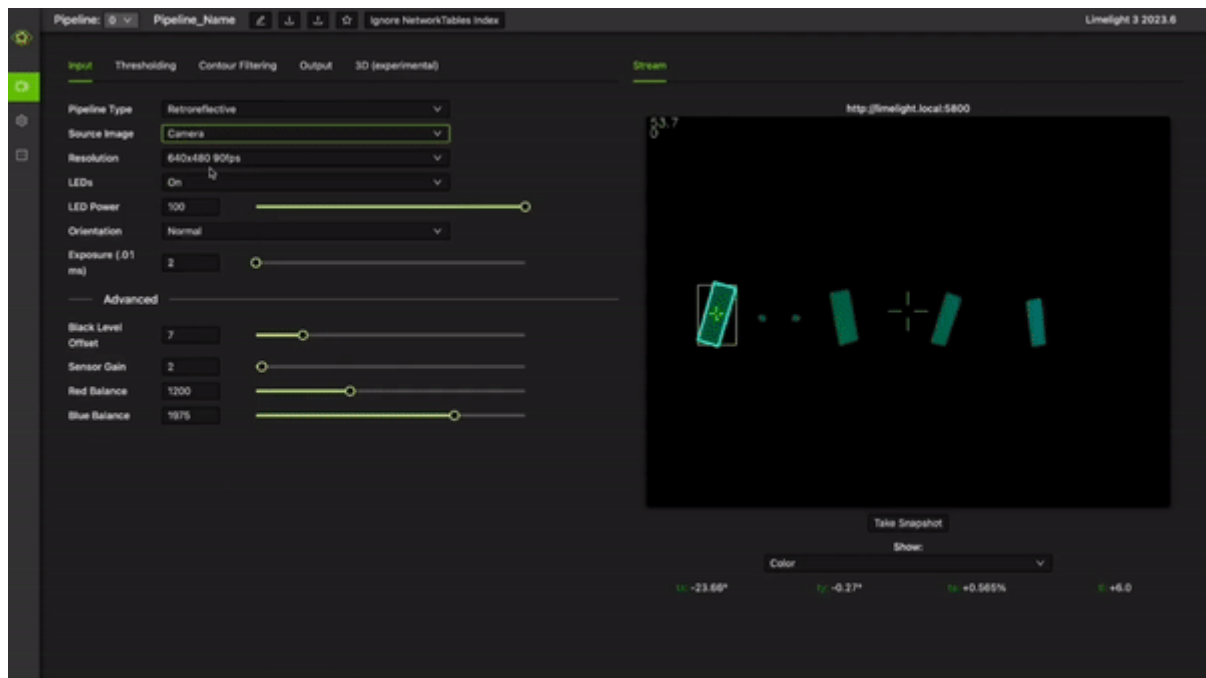
Tipo da pipeline

Controla o tipo desejado de pipeline (AprilTags, Redes Neurais, Python, etc.).

Imagem de origem

Controla a fonte da imagem que é passada pela pipeline. Mude para "Snapshot" para testar suas pipelines de visão em snapshots armazenados.

Este controle é redefinido automaticamente para "Camera" quando a interface gráfica é fechada.



Resolução + zoom

Controla a resolução da câmera e da pipeline de visão. Recomendamos o uso da pipeline 320x240, a menos que você esteja utilizando funcionalidades 3D.

Pipelines de 320x240 executam a 90fps, enquanto pipelines de 960x720 executam a 22 fps. Em 2020, foram adicionadas opções de Zoom de Hardware 2x e 3x para este campo. As opções de zoom não são digitais e usam pixels reais do sensor a 100%.

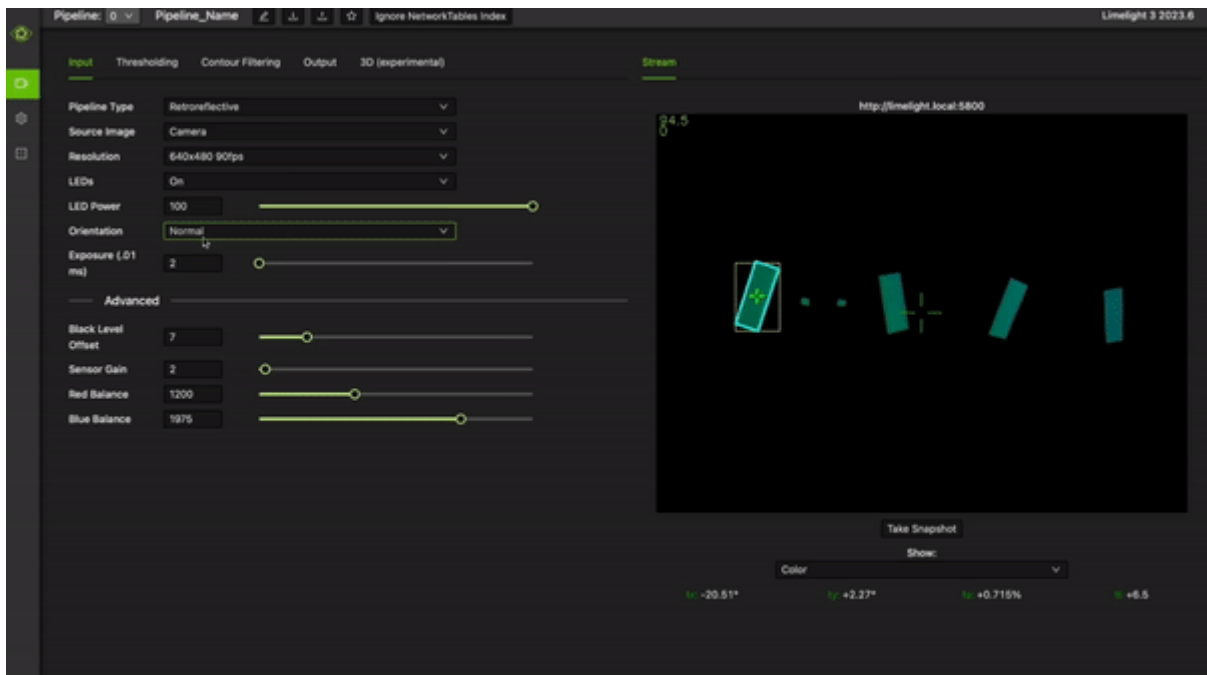
LEDs

Controla o modo padrão de LED para esta pipeline. Isso pode ser substituído durante uma partida com a opção de tabela de rede "LED".

Usuários do Limelight 2+ têm acesso a um controle deslizante de "Brilho do LED" que permite ajustar o brilho dos LEDs.

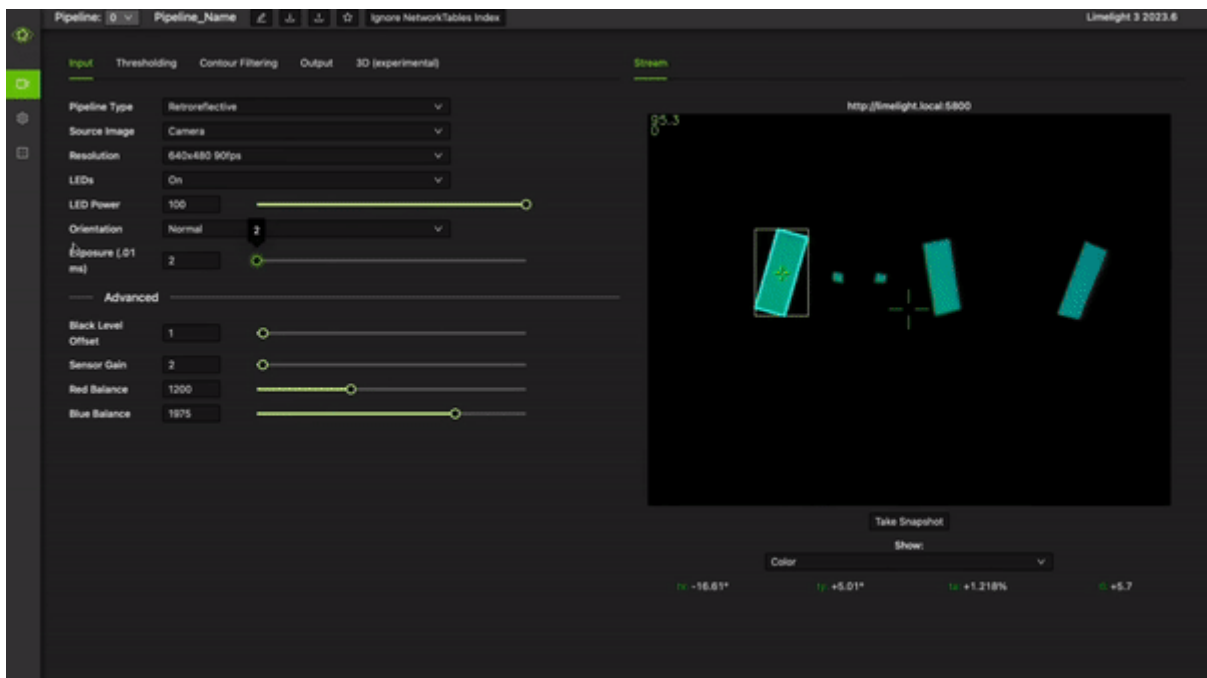
Orientação

Controla a orientação dos frames de entrada. Defina como "inverted" se sua câmera estiver montada de cabeça para baixo.



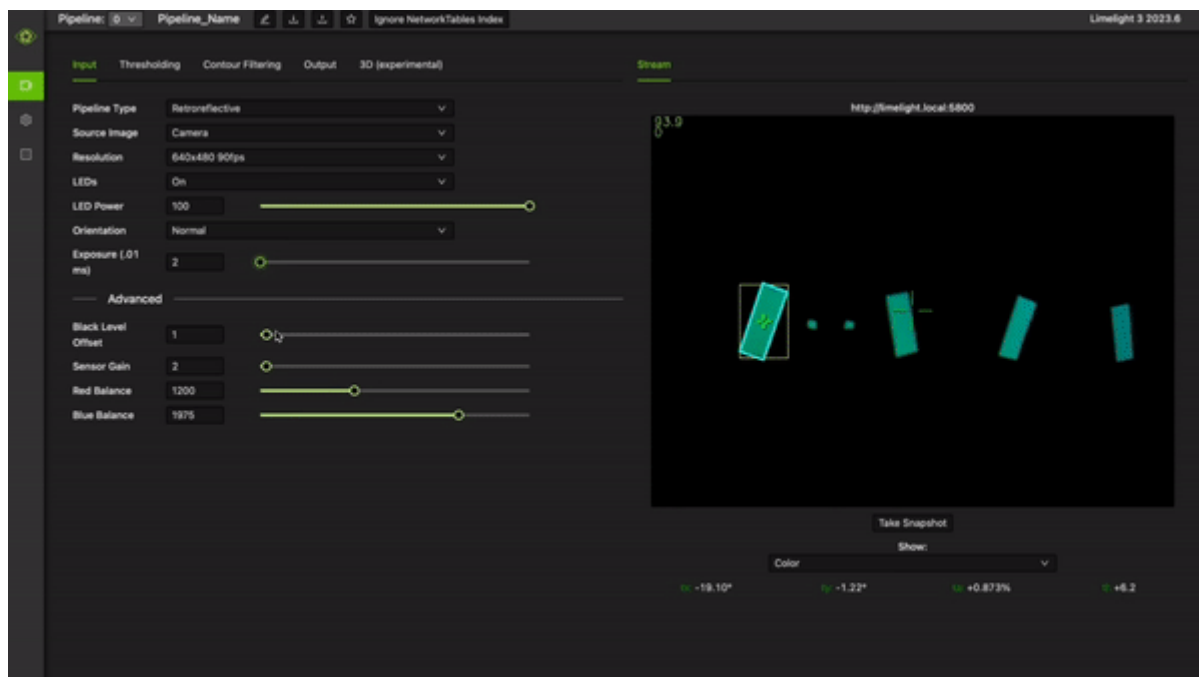
Exposição

Controla a configuração de exposição da câmera em intervalos de 0,01 milissegundos. Pense em uma câmera como uma grade de "buckets" (recipientes) para coletar luz - o tempo de exposição controla por quanto tempo os "buckets" da sua câmera ficam abertos por quadro. Reduzir o tempo de exposição efetivamente escurecerá sua imagem. Tempos de exposição baixos e fixos são cruciais no FRC, pois escurecem a maior parte dos dados de imagem recebidos. Fitas retrorrefletivas bem iluminadas se destacarão em uma imagem principalmente escura, tornando o processamento de visão um processo direto.



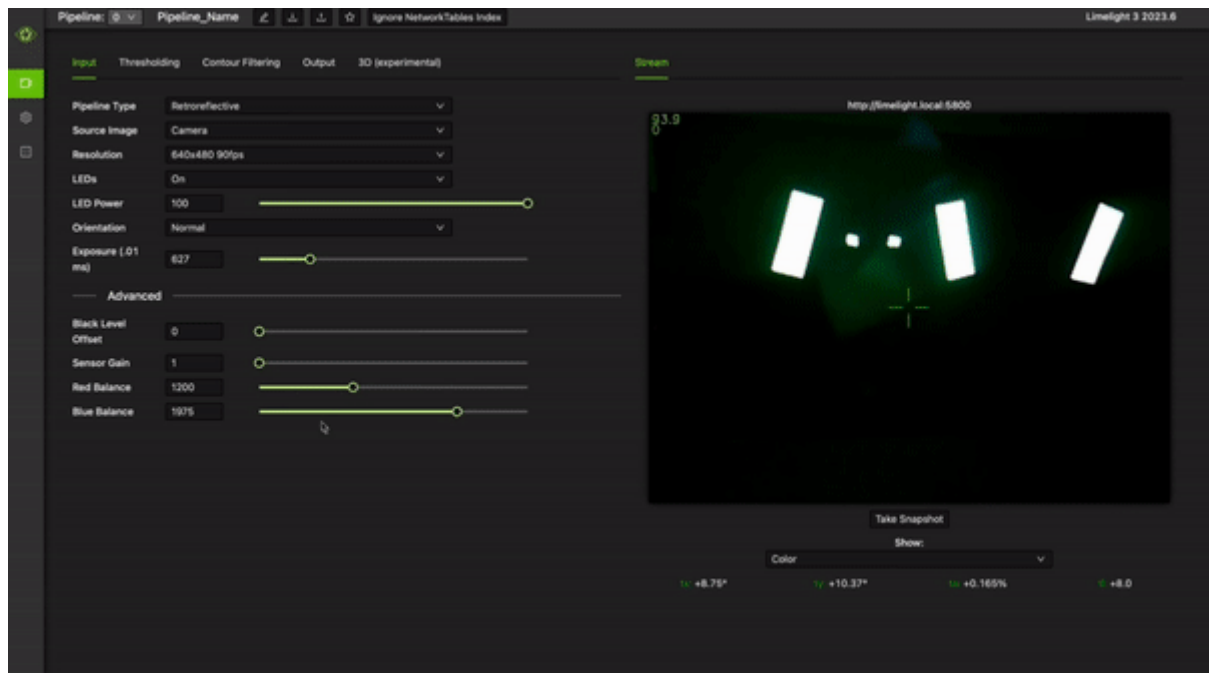
Deslocamento do nível de preto

Aumentar o deslocamento do nível de preto pode escurecer significativamente o stream da sua câmera. Isso deve ser aumentado para remover ainda mais as luzes da arena e pontos brilhantes da sua imagem. Esta é uma configuração de nível de sensor e não uma configuração de brilho digital falso.



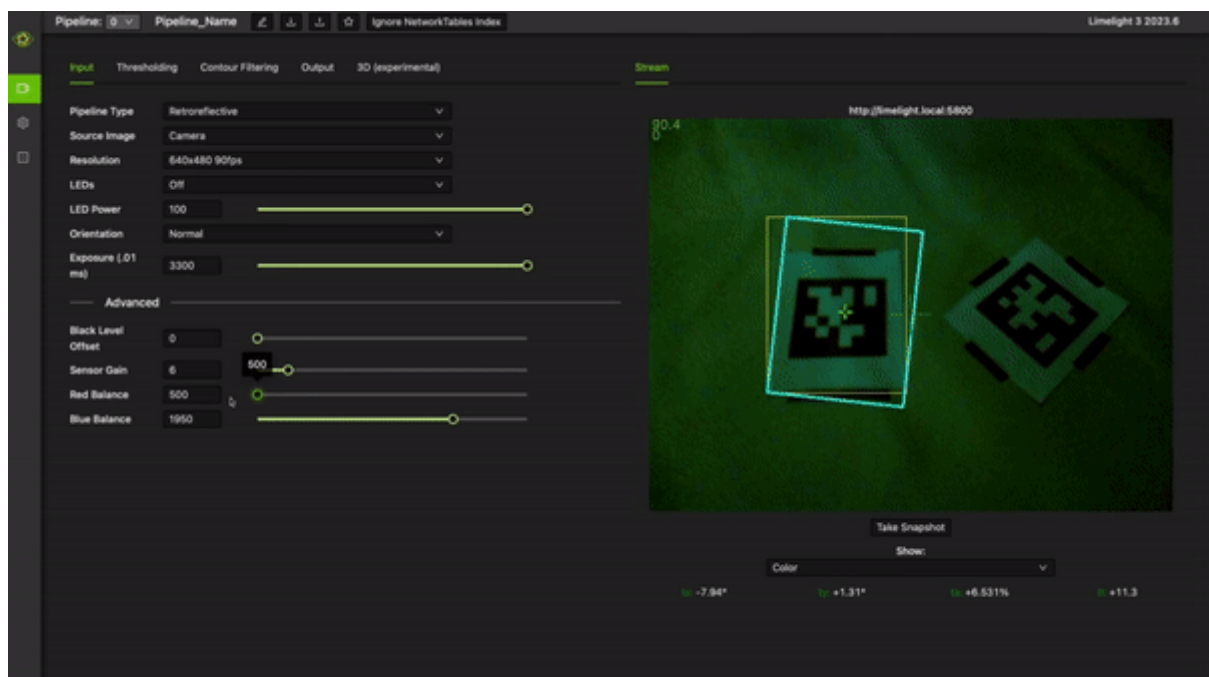
Ganho do Sensor

Aumentar o ganho do sensor efetivamente aumentará o brilho da imagem, mas geralmente adicionará ruído à imagem também. Você pode usar o Ganho do Sensor e o Deslocamento do Nível de Preto para iluminar a imagem sem aumentar o tempo de exposição. Isso minimizará o desfoque de movimento para aplicações de rastreamento em alta velocidade.



Balanco de Vermelho, Balanco de Azul

Controla a intensidade dos componentes de cor vermelha e azul na sua imagem. Isso controla coletivamente o balanço de branco do seu Limelight. Recomendamos deixar esses valores inalterados.



Pipeline de AprilTags

Rastreando AprilTags

As AprilTags são rastreadas usando os valores "tx", "ty" e "ta" no NetworkTables, da mesma forma que os alvos retro-refletivos padrão! Nenhuma alteração de código é necessária para atualizar um robô de rastreamento retro-refletivo para AprilTags. "botpose" e "campose" também podem ser usados para rastreamento 3D no espaço de campo e no espaço do alvo.

Para um uso mais avançado com várias tags, pode-se usar o despejo de resultados em formato JSON.

Não se sinta pressionado a usar os recursos mais avançados nas páginas "Avançado" a menos que saiba que precisa deles. Muitas das melhores equipes na FRC usam as técnicas mais simples disponíveis para maximizar confiabilidade e velocidade. Se você frequenta o Discord, CD e regionais com equipes de elite, pode ter a impressão de que precisa do software mais avançado possível para vencer eventos, mas isso simplesmente não é verdade.

Nossa mensagem para muitas das equipes que ajudamos é "Está tudo bem fazer algo simples."

Início rápido de AprilTags para FRC

- Guia de Entrada - Altere o *Pipeline Type* para "Fiducial Markers".
- Guia Padrão - Certifique-se de que a *family* esteja configurada como *AprilTag Classic 16h5*.
- Guia de Entrada - Defina o *Black Level* como zero.
- Guia de Entrada - Defina o *Gain* como 15.
- Guia de Entrada - Reduza a exposição para diminuir o desfoque de movimento. Pare de reduzir quando a confiabilidade do rastreamento diminuir.
- Guia Padrão - Se desejar aumentar a taxa de quadros, aumente a *Detector Downscale*.
- Guia de Entrada - Para aumentar o alcance e/ou a precisão, aumente a resolução de captura.
- Se estiver vendo detecções de tag espúrias, adicione os IDs que deseja rastrear ao controle *filter* ou aumente o valor do *Quality Threshold*.
- Clique no ícone de engrenagem e certifique-se de que o número da sua equipe e um IP estático estejam configurados.
- Clique em *Change Team Number* e *Change IP Settings* se você alterou essas configurações correspondentes. Reinicie seu robô.
- Você terminou! Use "tx" e "ty" na NetworkTables. Copie o exemplo de código na página "Iniciando".

Dicas

Para um rastreamento ideal, considere o seguinte:

- Suas tags devem estar o mais planas possível.
- Seu Limelight deve ser montado acima ou abaixo da altura da tag e inclinado para cima/baixo de forma que o alvo esteja centralizado. Seu alvo deve parecer o mais trapezoidal possível da perspectiva da sua câmera. Não é desejável que sua câmera esteja completamente "de frente" para uma tag se você quiser evitar inversões de tag.

Existe uma interação entre as seguintes variáveis para o rastreamento de AprilTag:

- Aumentar a resolução de captura sempre aumentará a precisão 3D e a estabilidade 3D. Isso também reduzirá a taxa de inversões de ambiguidade na maioria das perspectivas. Geralmente, aumentará o alcance. No entanto, isso reduzirá a taxa de quadros da linha de processamento.
- Aumentar a redução do detector sempre aumentará a taxa de quadros da linha de processamento. Isso diminuirá o alcance efetivo, mas, em alguns casos, isso pode ser negligenciável. Não afetará a precisão 3D, a estabilidade 3D ou a precisão na decodificação.
- Reduzir a exposição sempre melhorará a resistência ao desfoque de movimento. Isso é fácil de observar. Isso pode reduzir o alcance.
- Reduzir o brilho e contraste da imagem geralmente melhorará a taxa de quadros da linha de processamento e reduzirá o alcance.
- Aumentar o ganho do sensor permite aumentar o brilho sem aumentar a exposição. Isso pode reduzir a estabilidade 3D e a estabilidade do rastreamento.

Guia de Entrada

A guia de entrada hospeda controles para alterar a imagem bruta da câmera antes que ela passe pelo pipeline de processamento. Consulte a página "Construindo um pipeline retrorefletor/cor" para obter mais detalhes.

Para rastrear AprilTags:

- Altere o "Tipo de Pipeline" para "Marcadores Fiduciais".
- Defina o "Nível de Preto" como zero.

Neste ponto, é uma questão de equilibrar o ganho do sensor e o tempo de exposição. Você deseja ser capaz de ver as tags com a menor exposição possível para minimizar o desfoque de movimento. Isso geralmente requer uma configuração de ganho de sensor alta. Para um rastreamento simples em 2D, geralmente é aconselhável maximizar o ganho do sensor e, em seguida, aumentar a exposição a partir de zero até que os alvos sejam rastreados corretamente. Certifique-se de que a família correta esteja selecionada na guia *Standard* se o rastreamento não

estiver funcionando.

Guia Padrão

Family

Selecione o tipo de família *fiducial*/AprilTags. Para FRC você deve escolher o tipo 36h11.

Marker Size

Defina o tamanho esperado das *tags*, em mm, que seu robô vai encontrar. Para FRC isso deve ser definido como 165.1 (152.4 para *tags* de 2023)

Detector Downscale

Aumentar esse número resulta em um grande ganho de performance. Algumas vezes isso vai acarretar em um baixo alcance, mas é mínimo.

ID Filters

Filtros de ID permitem que você especifique exatamente quais *tags* são relevantes. Para a maioria das equipes da FRC, cada pipeline deve ser configurado para rastrear exatamente um ID de tag. Isso é feito por meio de uma lista separada por vírgulas de números (por exemplo, "0,1"). Essa funcionalidade é importante para eliminar a grande maioria dos falsos positivos.

<https://player.vimeo.com/video/872728604?h=ac074f2f00>

Cropping

O recorte remove conteúdo da imagem para melhorias significativas de desempenho. Utilize a chave "crop" do NetworkTables para recortar dinamicamente durante as partidas.

<https://player.vimeo.com/video/872728905?h=87c4609a8b>

Multi-Target Sorting and Grouping

Isso permite a funcionalidade exata de agrupamento vista em pipelines retro-refletivos padrão. Na maioria dos jogos, a única característica a ser modificada é o filtro "Área", que permitirá que você exclua *tags* pequenas.

<https://player.vimeo.com/video/872752999?h=c9b5c5dd9f>

AprilTags 3D

Existem três níveis de rastreamento 3D de AprilTag no Limelight OS:

- Rastreamento de ponto de interesse (Fácil de usar, não requer alterações de código, compatível com "tx" e "ty").
- Rastreamento 3D completo.
- Localização do robô.
- Todos os dados 3D são acessíveis diretamente via NetworkTables ou JSON.

Rastreamento do ponto de interesse

O rastreamento de ponto de interesse permite que você defina um ponto de interesse 3D em relação a uma AprilTag.

Suponha que você esteja tentando mirar em uma parte do campo que está a 6 polegadas à esquerda e 2 polegadas de uma AprilTag. Você pode simplesmente definir esse ponto de interesse na interface web (em metros) e, em seguida, rastrear esse ponto 3D usando tx e ty como se ele existisse como um alvo no mundo real.

<https://player.vimeo.com/video/872728776?h=a2e49afc44>

Rastreamento 3D completo

O rastreamento 3D completo é acessível por meio da matriz *campose* na tabela de NetworkTables e por meio da saída de resultados em JSON. Na seção "visualizer" na guia *Advanced*, você encontrará vários visualizadores que ajudarão a entender o propósito de cada uma das transformações disponíveis no despejo JSON. Em geral, as transformações mais úteis serão "Transformação da Câmera no Espaço do Alvo" e "Transformação do Robô no Espaço do Alvo". Consulte a documentação do sistema de coordenadas para obter mais detalhes.

Observe os ajustes de pose do Limelight (espaço do robô) nesta demonstração:

<https://player.vimeo.com/video/872728626?h=6142d91c50>

Localização do robô com MegaTags

Se a pose do robô no espaço Limelight foi configurada na interface web e um mapa de campo foi carregado através da interface web, então a localização do robô no espaço de campo estará disponível através do array "botpose" nas networktables (x, y, z em metros, roll, pitch, yaw em graus).

Nossa implementação de botpose é chamada MegaTag. Se mais de uma tag estiver em vista, ela é resiliente a ambiguidades individuais das tags e ruídos na imagem. Se todos os keypoints estiverem coplanares, ainda há algum risco de ambiguidade de inversão.

- Cilindro Verde: Pose do robô individual por tag
- Cilindro Azul: BotPose Antigo
- Cilindro Branco: MegaTag Botpose

<https://player.vimeo.com/video/872728585?h=b621ab100a>

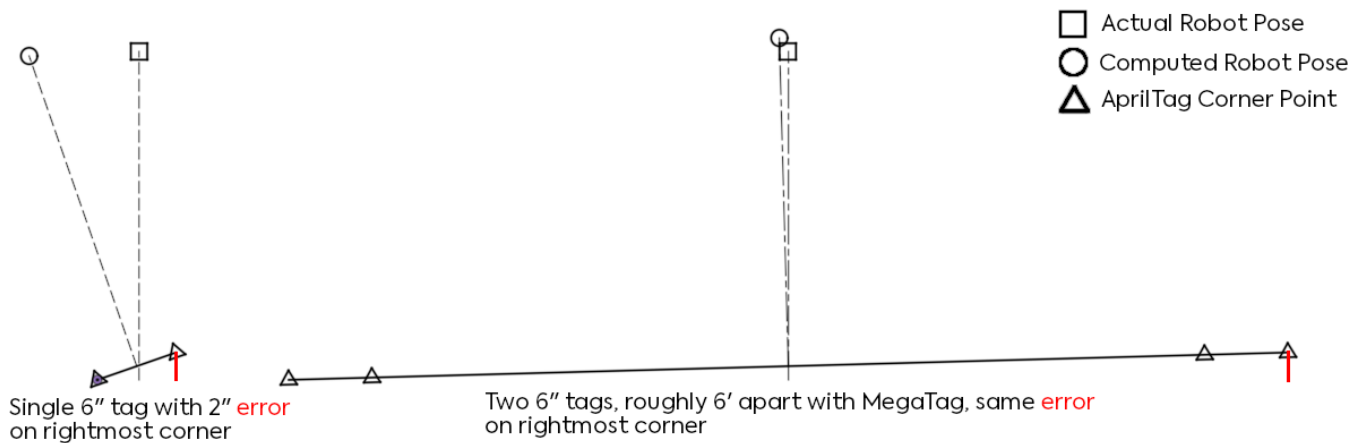
Note a pose ambígua aqui:

<https://player.vimeo.com/video/872728564?h=3e77d047a3>

Observe como a nova pose do robô (cilindro branco) é extremamente estável em comparação com a pose antiga do robô (cilindro azul). Você pode observar os valores de tx e ty também.

Isso não se limita a tags planas. Ele se adapta a qualquer número de tags em 3D completo e em qualquer orientação. Tags no chão e no teto funcionam perfeitamente.

Aqui está um diagrama que demonstra um aspecto de como isso funciona com um caso simples e plano. Os resultados são realmente melhores do que o que é mostrado, já que o MegaTag representado tem um erro significativo aplicado a três pontos em vez de um. Conforme o MegaTag 3D combinado aumenta em tamanho e no número de keypoints, sua estabilidade aumenta.



Utilizando Estimador de Pose da WPILib

As últimas imagens para o Limelight publicam a latência de mira e a latência de captura em milissegundos. Você pode acessá-las com as chaves NT "tl" e "cl", ou com `LimelightHelpers.getLatency_Pipeline()` e `LimelightHelpers.getLatency_Capture()` se estiver usando a biblioteca Limelight. Você também pode obter a latência combinada acessando o sétimo valor no array `botpose`.

psuedo-código para componente de latência do `addVisionMeasurement()` da WPILib:

```
Timer.getFPGATimestamp() - (tl/1000.0) - (cl/1000.0) or Timer.getFPGATimestamp() - (botpose[6]/1000.0)
```

Configurando a pose do robô da sua LimeLight

LL Forward, LL Right e LL Up representam distâncias ao longo dos vetores para a frente, para a direita e para cima do robô, se você fosse incorporar o robô (em metros). LL Roll, Pitch e Yaw representam a rotação do seu Limelight em graus. Você pode modificar esses valores e observar a mudança do modelo 3D do Limelight no visualizador 3D. O Limelight utiliza essa configuração internamente para transitar da pose alvo no espaço da câmera para a pose do robô no espaço do campo.

Sistema de coordenadas 3D em detalhes

Espaço da câmera da LimeLight

Sistema de Coordenadas Cartesianas 3D com $(0,0,0)$ na lente da câmera.

X+ → Apontando para a direita (se você estivesse incorporando a câmera)

Y+ → Apontando para baixo

Z+ → Apontando para fora da câmera

Espaço do alvo

Sistema de coordenadas cartesianas 3D com $(0, 0, 0)$ no centro do alvo

X+ → Apontando para a direita do alvo (se você estiver olhando para o alvo)

Y+ → Apontando para baixo

Z+ → pontando para fora do alvo (ortogonal ao plano do alvo)

Espaço do robô

Sistema de Coordenadas Cartesianas 3D com $(0,0,0)$ localizado no centro do quadro do robô projetado para baixo até o chão.

X+ → Apontando para frente (Vetor para a Frente)

Y+ → Apontando para a direita do robô (Vetor para a Direita)

Z+ → Apontando para cima (Vetor para Cima)

Espaço do campo

Sistema de Coordenadas Cartesianas 3D com (0,0,0) localizado no centro do campo.

X+ → Aponta ao longo do lado longo do campo

Y+ → Aponta para cima ao longo do lado curto do campo

Z+ → Aponta em direção ao céu

Destro. Um ângulo theta positivo resulta em uma rotação no sentido anti-horário a partir da perspectiva positiva externa.

Especificação do mapa AprilTag

A funcionalidade de localização no espaço do campo do Limelight utiliza arquivos .fmap para calcular uma pose do robô a ser usada pelos estimadores de pose do WPILIB. Nossos arquivos .fmap suportam mapas compostos por diferentes tamanhos de alvo e diferentes famílias.

Você pode usar os .fmap para definir "ambientes", como campos de competição FRC, ou "objetos", como objetos que possuem vários AprilTags anexados. Para usar um .fmap, tudo o que você precisa fazer é fazer o upload para o seu Limelight usando a interface ou uma das APIs de upload (em breve).

O arquivo .fmap é um arquivo JSON contendo uma única matriz "fiducial". Cada entrada na matriz fiducial tem a seguinte estrutura:

family	AprilTag/Família Fiducial
id	Id da Tag
size	Tamanho da Tag em mm
transform	Transformação de matriz 4x4 do alvo, Linha-Maior, unidades no SI
unique	Especifica se o alvo é único neste mapa ou se aparece várias vezes

Mapa de AprilTag para FRC Charged Up 2023

```

{

```

```
"fiducials": [  
  {  
    "family": "apriltag3_16h5_classic",  
    "id": 1,  
    "size": 152.4,  
    "transform": [  
      -1,  
      0,  
      0,  
      7.24310,  
      0,  
      -1,  
      0,  
      -2.93659,  
      0,  
      0,  
      1,  
      0.46272,  
      0,  
      0,  
      0,  
      1  
    ],  
    "unique": 1  
  },  
  {  
    "family": "apriltag3_16h5_classic",  
    "id": 2,  
    "size": 152.4,  
    "transform": [  
      -1,  
      0,  
      0,  
      7.24310,  
      0,  
      -1,  
      0,  
      -1.26019,  
      0,  
      0,  
      0
```

```
1,
0.46272,
0,
0,
0,
1
],
"unique": 1
},
{
"family": "apriltag3_16h5_classic",
"id": 3,
"size": 152.4,
"transform": [
-1,
0,
0,
7.24310,
0,
-1,
0,
0.41621,
0,
0,
1,
0.46272,
0,
0,
0,
1
],
"unique": 1
},
{
"family": "apriltag3_16h5_classic",
"id": 4,
"size": 152.4,
"transform": [
-1,
0,
```

```
0,
7.90832,
0,
-1,
0,
2.74161,
0,
0,
1,
0.695452,
0,
0,
0,
1
],
"unique": 1
},
{
"family": "apriltag3_16h5_classic",
"id": 5,
"size": 152.4,
"transform": [
1,
0,
0,
-7.90832,
0,
1,
0,
2.74161,
0,
0,
1,
0.695452,
0,
0,
0,
1
],
"unique": 1
```

```
},
{
  "family": "apriltag3_16h5_classic",
  "id": 6,
  "size": 152.4,
  "transform": [
    1,
    0,
    0,
    -7.24310,
    0,
    1,
    0,
    0.41621,
    0,
    0,
    1,
    0.46272,
    0,
    0,
    0,
    1
  ],
  "unique": 1
},
{
  "family": "apriltag3_16h5_classic",
  "id": 7,
  "size": 152.4,
  "transform": [
    1,
    0,
    0,
    -7.24310,
    0,
    1,
    0,
    -1.26019,
    0,
    0,
```

```
    1,
    0.46272,
    0,
    0,
    0,
    1
  ],
  "unique": 1
},
{
  "family": "apriltag3_16h5_classic",
  "id": 8,
  "size": 152.4,
  "transform": [
    1,
    0,
    0,
    -7.24310,
    0,
    1,
    0,
    -2.93659,
    0,
    0,
    1,
    0.46272,
    0,
    0,
    0,
    1
  ],
  "unique": 1
}
]
```

```
}
```

Pipeline de AprilTags

Ferramenta de construção de mapa

A [ferramenta de construir mapa](#) é um editor 3D completo para mapas de AprilTags.

Pipelines de Redes Neurais

Começando com Redes Neurais

Com os pipelines de redes neurais do Limelight, desafios de visão computacional que antes eram considerados impossíveis agora são triviais. A visão baseada em aprendizado já desempenha um papel enorme em robôs de última geração e veículos autônomos, então estamos empolgados em trazer essa tecnologia para os estudantes da FIRST.

O Limelight 1, 2 e 3 todos suportam visão baseada em aprendizado com a ajuda do Google Coral. O Google Coral é um acelerador USB que deve ser adquirido separadamente.

Veja o campeão do mundial 1323 de 2023 usando redes neurais da limelight:

<https://www.youtube.com/embed/hz3MolcRe2M?si=fMim1DUpanTnYqUy>

Baixe redes neurais pré-treinadas em nossa página de [downloads](#).

Na FRC, equipes sempre quiseram rastrear peças de jogo no campo durante os períodos autônomo e teleoperado. Usando o pipeline "Neural Detector" do Limelight, equipes podem rastrear peças da mesma forma que qualquer outro alvo sem ajustes.

Os pipelines "Neural Classifier", por outro lado, permitem que as equipes adicionem capacidades avançadas de detecção aos seus robôs. Digamos que uma equipe queira determinar se seu robô está com uma bola vermelha, uma bola azul ou sem bola. Um Limelight apontado para dentro de um robô poderia executar um classificador treinado para determinar um desses três casos. Um classificador também poderia contar o número de objetos em uma comporta, determinar o estado de uma característica no campo, etc.

Redes de Detector e Classificador Neural requerem a adição de um acelerador USB Google Coral. O Acelerador Google Coral é um ASIC (circuito integrado específico para aplicação) projetado especificamente para inferência de redes neurais. Você pode pensar no termo "inferência" como "execução" ou "processamento de dados através da rede neural para produzir uma saída".

Se você estiver interessado em construir uma compreensão mais profunda de aprendizado de máquina, recomendamos começar com este vídeo do [3blue1brown](#).

Programadores podem aprender mais de maneira prática com o [seguinte livro](#)

Pipeline de Detector Neural

Para começar, certifique-se de que seu Google Coral está conectado à porta USB-A no seu Limelight.

Altere o *Type Pipeline* para *Neural Detector* para começar a executar o modelo de teste incorporado. Baixe redes neurais pré-treinadas na página de [downloads](#) e faça o upload para começar a rastrear peças de jogo.

Ajuste o controle deslizante *confidence threshold* para alterar a confiança necessária para uma detecção bem-sucedida. Todos os resultados são enviados por JSON, mas recomendamos usar a interface de classificação incorporada para otimizar para um único alvo, que será representado pelos valores das networktables "tx," "ty," "ta," e "tclass."

Altere a janela de recorte para ignorar facilmente objetos fora da zona de detecção desejada.

<https://player.vimeo.com/video/872728678?h=cb10161117>

Pipeline de classificador neural

Para começar, certifique-se de que seu Google Coral está conectado à porta USB-A no seu Limelight.

Altere o *Pipeline Type* para *Neural Classifier* para começar a executar inferência no modelo de teste incorporado. Você pode treinar seus próprios modelos de classificador usando o método documentado na seção "Treinamento".

A janela de *Crop* permitirá que você tenha um controle melhor sobre a imagem usada para inferência da rede neural. Embora os modelos de classificação sejam capazes de atingir níveis incríveis de generalização em ambientes diversos, você terá mais sucesso ao minimizar o número de variáveis na sua imagem.

Treinando um modelo de classificação customizado

Google Teachable Machine é uma ferramenta totalmente baseada em navegador que permite o usuário criar modelos de classificação sem código ou conhecimento avançado de *machine learning*.

Coleção de dados

Reúna imagens para cada classe que você deseja reconhecer. Certifique-se de ter um conjunto de dados equilibrado (um número aproximadamente igual de imagens para cada classe). Garanta diversidade em ângulos de câmera, iluminação, fundos, características do objeto, etc.

Treinando com teachable machine

Vá para o site Teachable Machine: [Teachable Machine](#). Clique no botão *Get Started*. Selecione *Image project* para criar um classificador de imagem. Para cada classe: clique no botão *Add Class*. Faça o upload das imagens relacionadas a essa classe. Após adicionar todas as classes, clique no botão *Train Model*. Uma vez que o treinamento estiver completo, você pode testar o modelo diretamente no navegador.

Treinando um modelo de detecção customizado

Com roboFlow, Google Colab, e seu próprio conjunto de dados, você pode rapidamente treinar seus modelos de detecção customizados para LimeLight.

Pipeline com Python

Pipelines com Python

Com scripts em Python, você pode aproveitar todo o poder do OpenCV para construir rapidamente seus próprios pipelines. O interpretador Python está integrado ao backend em C++ do Limelight, portanto, erros e falhas são tratados de maneira elegante.

Os pipelines regulares do SnapScript são programados diretamente na interface web do Limelight.

<https://player.vimeo.com/video/872718295?h=59f7302d19>

Limelight cuida do hardware, da interface com a câmera, da rede, da transmissão de dados e do pré-processamento básico de imagens. Tudo o que você precisa fazer é escrever uma única função em Python chamada `runPipeline()`.

- Uma das características mais importantes que oferecemos é o retículo (crosshair) de um clique. O retículo, o retículo duplo, tx, ty, ta, ts, tvert e todas as outras leituras padrão do Limelight NetworkTables se ajustarão automaticamente ao contorno que você retornar da função Python `runPipeline()`.
- Escreva suas próprias visualizações em tempo real, limiares (thresholding), filtros e passe por completo pelo nosso backend se desejar.
 - O scripting Python do Limelight tem acesso às bibliotecas completas de OpenCV e numpy.
 - Além do acesso à imagem, a função `runPipeline()` também tem acesso à matriz numérica "llrobot" da NetworkTables. Envie quaisquer dados de seus robôs para seus scripts Python para visualização ou aplicações avançadas (pode-se enviar dados de IMU, dados de pose, velocidade do robô, etc., para uso em scripts Python).
 - A função `runPipeline` também produz uma matriz numérica que é colocada diretamente na matriz numérica "llpython" da NetworkTables. Isso significa que você pode ignorar completamente o retículo e outras funcionalidades do Limelight e enviar seus próprios dados personalizados de volta para seus robôs.
 - Os scripts Python são isolados dentro do nosso ambiente C++, então você não precisa se preocupar com falhas. Mudanças nos scripts são aplicadas instantaneamente, e quaisquer mensagens de erro são impressas diretamente na interface web.

```
import cv2
import numpy as np
```

```

# runPipeline() is called every frame by Limelight's backend.
def runPipeline(image, llrobot):
    # convert the input image to the HSV color space
    img_hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    # convert the hsv to a binary image by removing any pixels
    # that do not fall within the following HSV Min/Max values
    img_threshold = cv2.inRange(img_hsv, (60, 70, 70), (85, 255, 255))

    # find contours in the new binary image
    contours, _ = cv2.findContours(img_threshold,
    cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    largestContour = np.array([[]])

    # initialize an empty array of values to send back to the robot
    llpython = [0,0,0,0,0,0,0,0]

    # if contours have been detected, draw them
    if len(contours) > 0:
        cv2.drawContours(image, contours, -1, 255, 2)
        # record the largest contour
        largestContour = max(contours, key=cv2.contourArea)

        # get the unrotated bounding box that surrounds the contour
        x,y,w,h = cv2.boundingRect(largestContour)

        # draw the unrotated bounding box
        cv2.rectangle(image,(x,y),(x+w,y+h),(0,255,255),2)

        # record some custom data to send back to the robot
        llpython = [1,x,y,w,h,9,8,7]

    #return the largest contour for the LL crosshair, the modified image, and custom robot data
    return largestContour, image, llpython

```

Pro Pipelines com Python

Os pipelines SnapScript Pro são programados no VSCode e implantados por meio do plugin Limelight VSCode. Se o seu pipeline precisar de vários arquivos Python, arquivos de ativos adicionais, ou se você quiser garantir que seu código Python esteja em controle de origem, os pipelines SnapScript Pro oferecem o nível perfeito de personalização.

API's

NetworkTables API completa

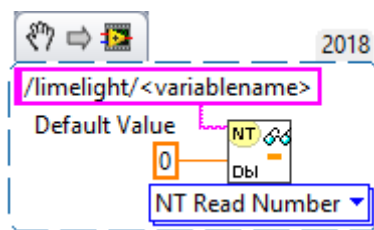
Dados básicos de segmento

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

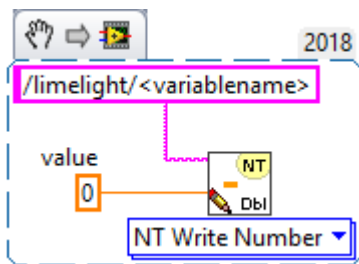
Controle de câmera

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

Aviso

A documentação oficial da Limelight agora tem tradução para o português!

[Acesse aqui!](#)