GrabCut+D

Fabian Andres Prada Niño

Instituto Nacional de Matematica Pura e Aplicada

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.
- **Tools:**

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.
- **Tools:**

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.
- **Tools:**
  - GrabCut

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.
- **Tools:**
  - GrabCut
  - Kinect

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.
- **Tools:**
  - GrabCut
  - Kinect
- **Approaches:**

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.
- **Tools:**
  - GrabCut
  - Kinect
- **Approaches:**
  - First: Region Growing.

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.
- **Tools:**
  - GrabCut
  - Kinect
- **Approaches:**
  - First: Region Growing.
  - Second: Probabilistic Model.

Introduction

- **Goal:** Propose a robust method for object extraction in an RGBD image.
- **Tools:**
  - GrabCut
  - Kinect
- **Approaches:**
  - First: Region Growing.
  - Second: Probabilistic Model.
  - Third: Robust Depth Based Seeding

GraphCut

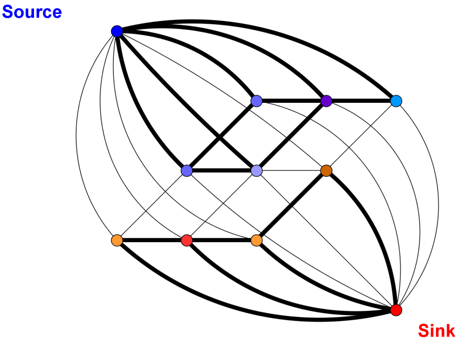
- Image segmentation method originally introduced by Boykov and Jolly [1].

GraphCut

- Image segmentation method originally introduced by Boykov and Jolly [1].
- Assigns to the image a weighted graph structure where each pixel is represented by a node.There are two additional nodes, source and sink, which represents Background(BG) and Foreground(FG) respectively:

GraphCut

- Image segmentation method originally introduced by Boykov and Jolly [1].
- Assigns to the image a weighted graph structure where each pixel is represented by a node.There are two additional nodes, source and sink, which represents Background(BG) and Foreground(FG) respectively:



Weighted Graph $\Leftrightarrow$ Energy Function

Weighted Graph $\Leftrightarrow$ Energy Function

$$E(x) = U(x, z, w) + V(x, z)$$

Weighted Graph $\Leftrightarrow$ Energy Function

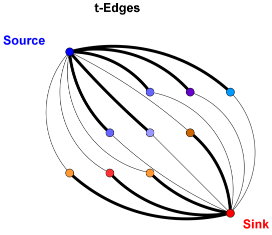
$$E(x) = U(x, z, w) + V(x, z)$$

- **Data Term:** $U(x, z, w)$. Measures the pixel's similarity to BG and FG models.

Weighted Graph ⇔ Energy Function

$$E(x) = U(x,z,w) + V(x,z)$$

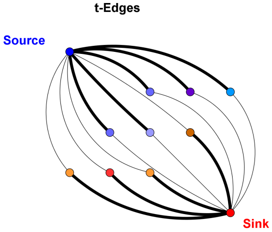
- **Data Term:** $U(x,z,w)$. Measures the pixel's similarity to BG and FG models.



Weighted Graph ⇔ Energy Function

$$E(x) = U(x, z, w) + V(x, z)$$

- **Data Term:** $U(x, z, w)$. Measures the pixel's similarity to BG and FG models.

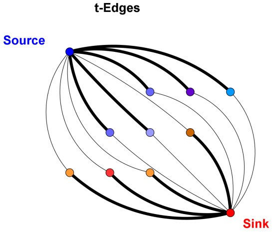


- **Smoothness Term:** $V(x, z)$. Measures the similarity of neighbours pixels.

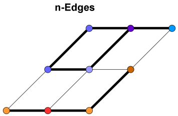
Weighted Graph ⇔ Energy Function

$$E(x) = U(x, z, w) + V(x, z)$$

- **Data Term:** $U(x, z, w)$. Measures the pixel's similarity to BG and FG models.



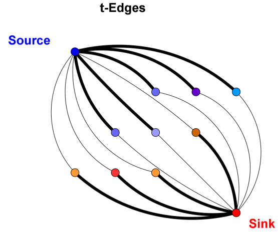
- **Smoothness Term:** $V(x, z)$. Measures the similarity of neighbours pixels.



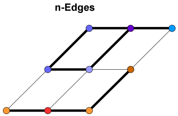
Weighted Graph ⇔ Energy Function

$$E(x) = U(x, z, w) + V(x, z)$$

- **Data Term:** $U(x, z, w)$. Measures the pixel's similarity to BG and FG models.



- **Smoothness Term:** $V(x, z)$. Measures the similarity of neighbours pixels.



GrabCut

GrabCut

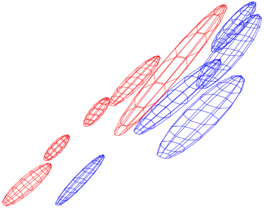
- Based in the principles of GraphCut, Rother, Kolmogorov and Blake proposed GrabCut [2].

GrabCut

- Based in the principles of GraphCut, Rother, Kolmogorov and Blake proposed GrabCut [2].
- Gasussian Mixture Models are built from the color data of certain pixels (**seeds**) to represent the color distribution of FG and BG.

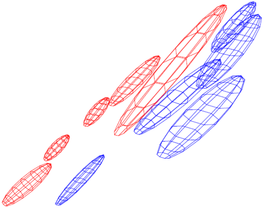
GrabCut

- Based in the principles of GraphCut, Rother, Kolmogorov and Blake proposed GrabCut [2].
- Gasussian Mixture Models are built from the color data of certain pixels (**seeds**) to represent the color distribution of FG and BG.



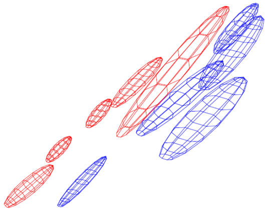
GrabCut

- Based in the principles of GraphCut, Rother, Kolmogorov and Blake proposed GrabCut [2].
- Gasussian Mixture Models are built from the color data of certain pixels (**seeds**) to represent the color distribution of FG and BG.



GrabCut

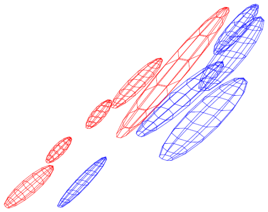
- Based in the principles of GraphCut, Rother, Kolmogorov and Blake proposed GrabCut [2].
- Gasussian Mixture Models are built from the color data of certain pixels (**seeds**) to represent the color distribution of FG and BG.



$$U(x, \omega_C, z) = -\alpha_C \left(\sum_p \log h_{BC}(z_p)[x_p = 0] + \log h_{FC}(z_p)[x_p = 1] \right)$$

GrabCut

- Based in the principles of GraphCut, Rother, Kolmogorov and Blake proposed GrabCut [2].
- Gasussian Mixture Models are built from the color data of certain pixels (**seeds**) to represent the color distribution of FG and BG.



$$U(x, \omega_C, z) = -\alpha_C \Big(\sum_p \log h_{BC}(z_p)[x_p = 0] + \log h_{FC}(z_p)[x_p = 1] \Big)$$

$$V(x, z, d) = \gamma_C \Big(\sum_{(p,q \in N)} dis(p, q)^{-1} (\exp\{-\beta_C ||z_p - z_q||^2\})[x_p \neq x_q] \Big)$$

GrabCut

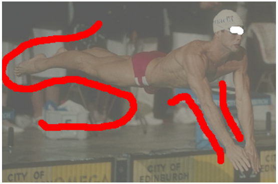
GrabCut

Initial Selection

GrabCut

Initial Selection

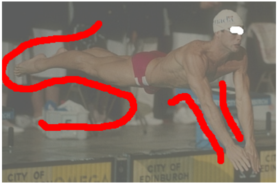
Further Interaction



GrabCut

Initial Selection

Further Interaction



Final Result

First Approach: Region Growing

First Approach: Region Growing

Main Idea: Extract an object from an image implementing a priority search algorithm to identify its connected-depth component.Then, apply GrabCut in the contour of the component to improve the result.

First Phase

First Phase

1. Pick an initial pixel in the interior of the FG object.

First Phase

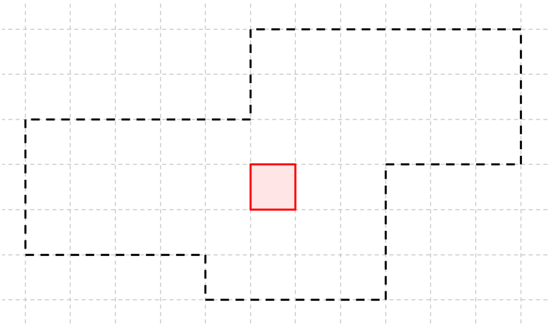
1. Pick an initial pixel in the interior of the FG object.
2. Fix parameters for global depth tolerance e_d and depth continuity e_c.

First Phase

1. Pick an initial pixel in the interior of the FG object.
2. Fix parameters for global depth tolerance e_d and depth continuity e_c.
3. BFS is implemented to identify the connected component of pixels satisfying $||d - d_0|| < e_d$ and $||d_{parent} - d_{pixel}|| < e_c$.

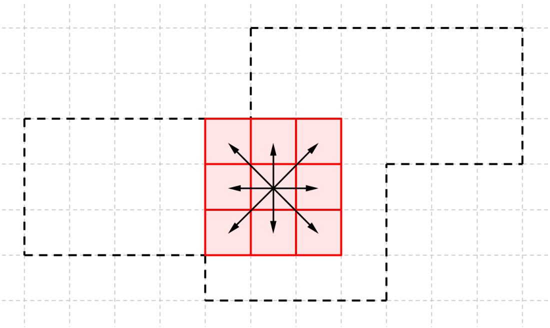
First Phase

1. Pick an initial pixel in the interior of the FG object.
2. Fix parameters for global depth tolerance e_d and depth continuity e_c.
3. BFS is implemented to identify the connected component of pixels satisfying $||d - d_0|| < e_d$ and $||d_{parent} - d_{pixel}|| < e_c$.



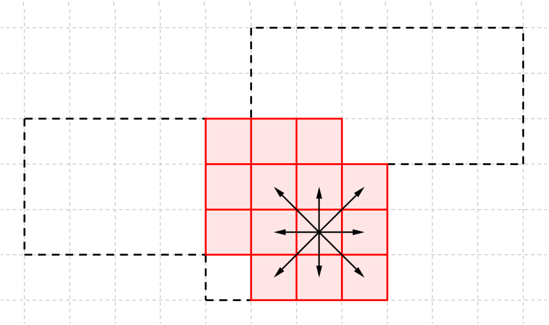
First Phase

1. Pick an initial pixel in the interior of the FG object.
2. Fix parameters for global depth tolerance e_d and depth continuity e_c.
3. BFS is implemented to identify the connected component of pixels satisfying $||d - d_0|| < e_d$ and $||d_{parent} - d_{pixel}|| < e_c$.



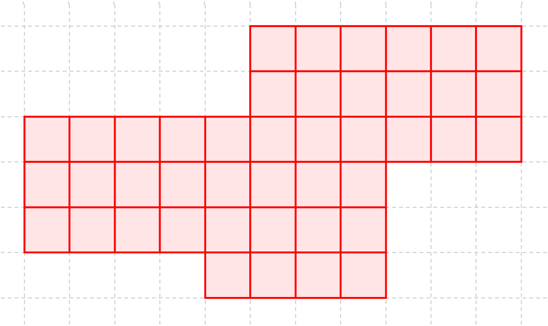
First Phase

1. Pick an initial pixel in the interior of the FG object.
2. Fix parameters for global depth tolerance e_d and depth continuity e_c.
3. BFS is implemented to identify the connected component of pixels satisfying $||d - d_0|| < e_d$ and $||d_{parent} - d_{pixel}|| < e_c$.



First Phase

1. Pick an initial pixel in the interior of the FG object.
2. Fix parameters for global depth tolerance e_d and depth continuity e_c.
3. BFS is implemented to identify the connected component of pixels satisfying $||d - d_0|| < e_d$ and $||d_{parent} - d_{pixel}|| < e_c$.



Second Phase

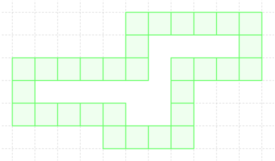
Second Phase

Second Phase

1. Define a band around the *contour pixels* of the component.

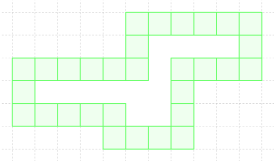
Second Phase

1. Define a band around the *contour pixels* of the component.



Second Phase

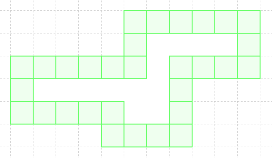
1. Define a band around the *contour pixels* of the component.



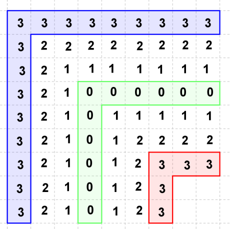
2. Pixels in the perimeter of the band belonging to the component are labelled as **FG seeds**, and those not belonging as **BG seeds**.

Second Phase

1. Define a band around the *contour pixels* of the component.

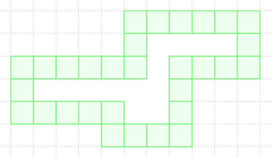


2. Pixels in the perimeter of the band belonging to the component are labelled as **FG seeds**, and those not belonging as **BG seeds**.

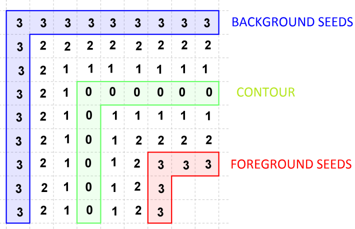


Second Phase

1. Define a band around the *contour pixels* of the component.



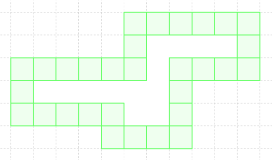
2. Pixels in the perimeter of the band belonging to the component are labelled as **FG seeds**, and those not belonging as **BG seeds**.



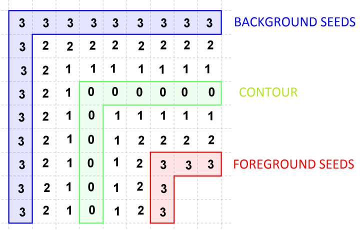
3. Apply Grabcut on the 8-connected graph associated to the band.

Second Phase

1. Define a band around the *contour pixels* of the component.



2. Pixels in the perimeter of the band belonging to the component are labelled as **FG seeds**, and those not belonging as **BG seeds**.



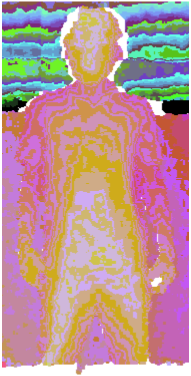
3. Apply Grabcut on the 8-connected graph associated to the band.

Results

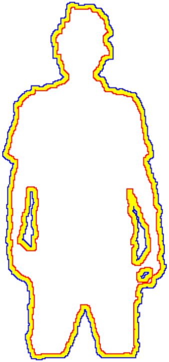
Results

RGB

Depth



Graph

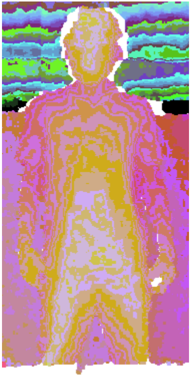


Result

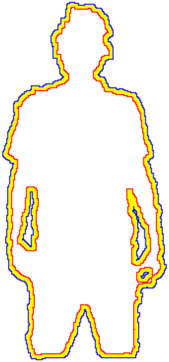
Results

RGB

Depth



Graph



Result

Depth data must be improved!

Temporal Filter

Temporal Filter

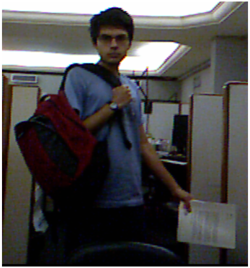
- Capture depth values for a sequence of frames.

Temporal Filter

- Capture depth values for a sequence of frames.
- For each pixel set the depth value to the mean of the non-zero samples.

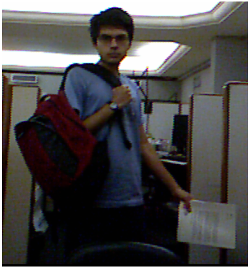
Temporal Filter

- Capture depth values for a sequence of frames.
- For each pixel set the depth value to the mean of the non-zero samples.



Temporal Filter

- Capture depth values for a sequence of frames.
- For each pixel set the depth value to the mean of the non-zero samples.



Color Based Depth Filter

Color Based Depth Filter

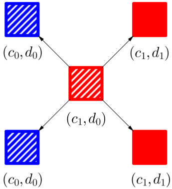
- Compared the depth value of each pixel with 4 near pixels

Color Based Depth Filter

- Compared the depth value of each pixel with 4 near pixels
- If there is depth disparity with at least two of them, then compare the colour values.

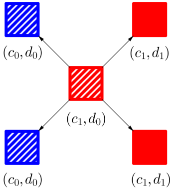
Color Based Depth Filter

- Compared the depth value of each pixel with 4 near pixels
- If there is depth disparity with at least two of them, then compare the colour values.



Color Based Depth Filter

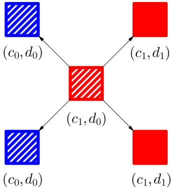
- Compared the depth value of each pixel with 4 near pixels
- If there is depth disparity with at least two of them, then compare the colour values.



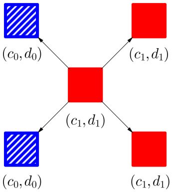
- Update the depth value of pixel if its colour is similar to pixels with different depth value, and its colour is different to pixels with similar depth value.

Color Based Depth Filter

- Compared the depth value of each pixel with 4 near pixels
- If there is depth disparity with at least two of them, then compare the colour values.

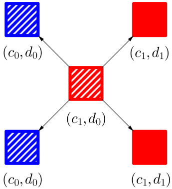


- Update the depth value of pixel if its colour is similar to pixels with different depth value, and its colour is different to pixels with similar depth value.

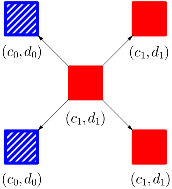


Color Based Depth Filter

- Compared the depth value of each pixel with 4 near pixels
- If there is depth disparity with at least two of them, then compare the colour values.



- Update the depth value of pixel if its colour is similar to pixels with different depth value, and its colour is different to pixels with similar depth value.



Result of Color Based Depth Filter

Result of Color Based Depth Filter

Filter

Previous Result

New Result

Detail

Remarks First Approach

Remarks First Approach

- Pros:

Remarks First Approach

- Pros:
  1. Simple user interaction.

Remarks First Approach

- Pros:
  1. Simple user interaction.
  2. Fast running time.

Remarks First Approach

- Pros:
  1. Simple user interaction.
  2. Fast running time.
  3. Accurate Colour Models

Remarks First Approach

- Pros:
  1. Simple user interaction.
  2. Fast running time.
  3. Accurate Colour Models
- Cons:

Remarks First Approach

- Pros:
    1. Simple user interaction.
    2. Fast running time.
    3. Accurate Colour Models
- Cons:
    1. Vulnerable to depth noise.

Remarks First Approach

- Pros:
  1. Simple user interaction.
  2. Fast running time.
  3. Accurate Colour Models
- Cons:
  1. Vulnerable to depth noise.
  2. No useful in the case of object-background adjacency.

Second Approach: Probabilistic Model

Second Approach: Probabilistic Model

Main Idea: Extend the energy function model implemented in GrabCut to include the depth data in analogous way to colour. Construct Gaussian Mixture Models associated to FG and BG depth and use them to define the seeding.

Involving depth data in Energy Function

Involving depth data in Energy Function

- Depth model ω_D is constructed from Gaussian Mixtures.

$$E(x, \omega_C, \omega_D, z, d) = U(x, \omega_C, \omega_D, z, d) + V(x, z, d)$$

Involving depth data in Energy Function

- Depth model ω_D is constructed from Gaussian Mixtures.

$$E(x, \omega_C, \omega_D, z, d) = U(x, \omega_C, \omega_D, z, d) + V(x, z, d)$$

- Parameters α_C and α_D assigns weight of colour and depth data.

Involving depth data in Energy Function

- Depth model ω_D is constructed from Gaussian Mixtures.

$$E(x, \omega_C, \omega_D, z, d) = U(x, \omega_C, \omega_D, z, d) + V(x, z, d)$$

- Parameters α_C and α_D assigns weight of colour and depth data.

$$\begin{aligned} U(x, \omega_C, \omega_D, z, d) = & -\sum_p \big(\alpha_C \log h_{BC}(z_p) + \alpha_D \log h_{BD}(d_p)\big)[x_p = 0] \\ & -\sum_p \big(\alpha_C \log h_{FC}(z_p) + \alpha_D \log h_{FD}(d_p)\big)[x_p = 1] \end{aligned}$$

Involving depth data in Energy Function

- Depth model ω_D is constructed from Gaussian Mixtures.

$$E(x, \omega_C, \omega_D, z, d) = U(x, \omega_C, \omega_D, z, d) + V(x, z, d)$$

- Parameters α_C and α_D assigns weight of colour and depth data.

$$\begin{aligned} U(x, \omega_C, \omega_D, z, d) = &- \sum_p \big(\alpha_C \log h_{BC}(z_p) + \alpha_D \log h_{BD}(d_p)\big)[x_p = 0] \\ &- \sum_p \big(\alpha_C \log h_{FC}(z_p) + \alpha_D \log h_{FD}(d_p)\big)[x_p = 1] \end{aligned}$$

- Parameters γ_C and γ_D assigns weight of colour and depth smoothness.

Involving depth data in Energy Function

- Depth model ω_D is constructed from Gaussian Mixtures.

$$E(x, \omega_C, \omega_D, z, d) = U(x, \omega_C, \omega_D, z, d) + V(x, z, d)$$

- Parameters α_C and α_D assigns weight of colour and depth data.

$$\begin{aligned} U(x, \omega_C, \omega_D, z, d) = & -\sum_p \big(\alpha_C \log h_{BC}(z_p) + \alpha_D \log h_{BD}(d_p)\big)[x_p = 0] \\ & -\sum_p \big(\alpha_C \log h_{FC}(z_p) + \alpha_D \log h_{FD}(d_p)\big)[x_p = 1] \end{aligned}$$

- Parameters γ_C and γ_D assigns weight of colour and depth smoothness.

$$\begin{aligned} V(x, z, d) = & \gamma_C \Big(\sum_{(p,q \in N)} dis(p,q)^{-1} \exp\{-\beta_C ||z_p - z_q||^2\}[x_p \neq x_q] \Big) + \\ & \gamma_D \Big(\sum_{(p,q \in N)} dis(p,q)^{-1} \exp\{-\beta_D ||d_p - d_q||^2\}[x_p \neq x_q] \Big) \end{aligned}$$

Involving depth data in Energy Function

- Depth model ω_D is constructed from Gaussian Mixtures.

$$E(x, \omega_C, \omega_D, z, d) = U(x, \omega_C, \omega_D, z, d) + V(x, z, d)$$

- Parameters α_C and α_D assigns weight of colour and depth data.

$$\begin{aligned} U(x, \omega_C, \omega_D, z, d) = & -\sum_p \big(\alpha_C \log h_{BC}(z_p) + \alpha_D \log h_{BD}(d_p)\big)[x_p = 0] \\ & -\sum_p \big(\alpha_C \log h_{FC}(z_p) + \alpha_D \log h_{FD}(d_p)\big)[x_p = 1] \end{aligned}$$

- Parameters γ_C and γ_D assigns weight of colour and depth smoothness.

$$\begin{aligned} V(x, z, d) = & \gamma_C \Big(\sum_{(p,q \in N)} dis(p, q)^{-1} \exp\{-\beta_C ||z_p - z_q||^2\}[x_p \neq x_q] \Big) + \\ & \gamma_D \Big(\sum_{(p,q \in N)} dis(p, q)^{-1} \exp\{-\beta_D ||d_p - d_q||^2\}[x_p \neq x_q] \Big) \end{aligned}$$

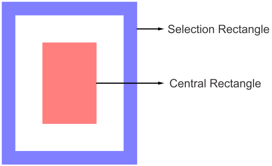
Seeding

Seeding

1. Construct BG Depth Model (BGDM) from the pixels in the border of the Selection Rectangle.

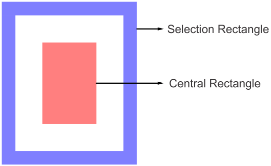
Seeding

1. Construct BG Depth Model (BGDM) from the pixels in the border of the Selection Rectangle.



Seeding

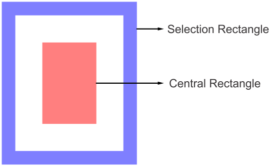
1. Construct BG Depth Model (BGDM) from the pixels in the border of the Selection Rectangle.



2. Construct FG Depth Model (FGDM) from the pixels in the interior of the Central Rectangle.

Seeding

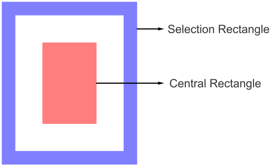
1. Construct BG Depth Model (BGDM) from the pixels in the border of the Selection Rectangle.



2. Construct FG Depth Model (FGDM) from the pixels in the interior of the Central Rectangle.
3. Discard the Gaussian components of FGDM whose mean value have the highest probability according to BGDM.

Seeding

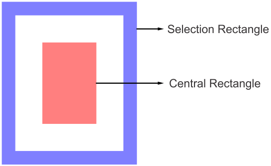
1. Construct BG Depth Model (BGDM) from the pixels in the border of the Selection Rectangle.



2. Construct FG Depth Model (FGDM) from the pixels in the interior of the Central Rectangle.
3. Discard the Gaussian components of FGDM whose mean value have the highest probability according to BGDM.
4. Evaluate all pixels in the image respect to these models, and mark them as FG Seeds or BG Seeds according to the obtain likelihood.

Seeding

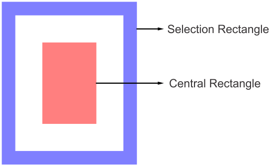
1. Construct BG Depth Model (BGDM) from the pixels in the border of the Selection Rectangle.



2. Construct FG Depth Model (FGDM) from the pixels in the interior of the Central Rectangle.
3. Discard the Gaussian components of FGDM whose mean value have the highest probability according to BGDM.
4. Evaluate all pixels in the image respect to these models, and mark them as FG Seeds or BG Seeds according to the obtain likelihood.
5. Build the definitive FGDM and FGCM from the FG Seeds. Do the same with BG Seeds.

Seeding

1. Construct BG Depth Model (BGDM) from the pixels in the border of the Selection Rectangle.



2. Construct FG Depth Model (FGDM) from the pixels in the interior of the Central Rectangle.
3. Discard the Gaussian components of FGDM whose mean value have the highest probability according to BGDM.
4. Evaluate all pixels in the image respect to these models, and mark them as FG Seeds or BG Seeds according to the obtain likelihood.
5. Build the definitive FGDM and FGCM from the FG Seeds. Do the same with BG Seeds.

Results

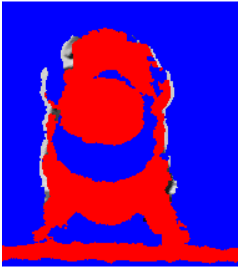
Results

Results

Results

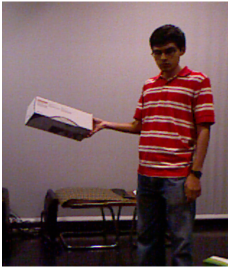
Results

Results



Results

Results



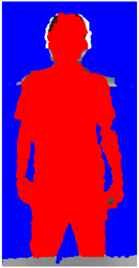
Results

Results

Results

Results



Results

Results

Results

Observations 1

Is the Gaussian Mixture Model convenient for depth data?

Observations 1

Is the Gaussian Mixture Model convenient for depth data?

Probably Not,...

Observations 1

Is the Gaussian Mixture Model convenient for depth data?

Probably Not,...

- In most cases depth range is greater in BG than FG.

Observations 1

Is the Gaussian Mixture Model convenient for depth data?

Probably Not,...

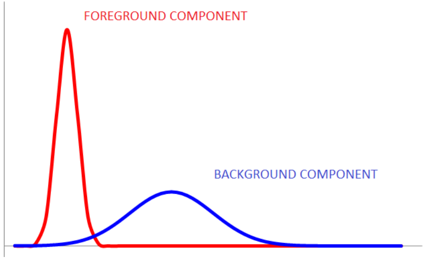
- In most cases depth range is greater in BG than FG.
- If we consider FGDM and BGDM with the same number of Gaussian components, the components in BGDM tend to have larger variance.

Observations 1

Is the Gaussian Mixture Model convenient for depth data?

Probably Not,...

- In most cases depth range is greater in BG than FG.
- If we consider FGDM and BGDM with the same number of Gaussian components, the components in BGDM tend to have larger variance.

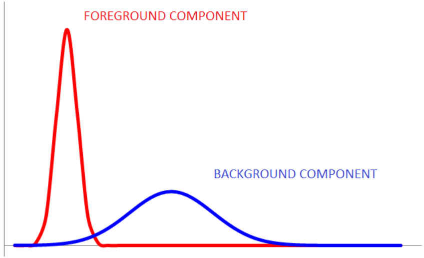


Observations 1

Is the Gaussian Mixture Model convenient for depth data?

Probably Not,...

- In most cases depth range is greater in BG than FG.
- If we consider FGDM and BGDM with the same number of Gaussian components, the components in BGDM tend to have larger variance.



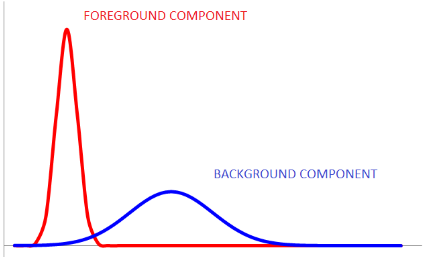
- If we increase the number of components in BGDM we improve in the variance problem, but we get BG clusters catching FG pixels (specially if object and background are close).

Observations 1

Is the Gaussian Mixture Model convenient for depth data?

Probably Not,...

- In most cases depth range is greater in BG than FG.
- If we consider FGDM and BGDM with the same number of Gaussian components, the components in BGDM tend to have larger variance.



- If we increase the number of components in BGDM we improve in the variance problem, but we get BG clusters catching FG pixels (specially if object and background are close).

Observations 2

Observations 2

How define a good set of Colour and Depth parameters?

Observations 2

How define a good set of Colour and Depth parameters?

- According to the depth contrast of the image the parameters require to be adjusted:

Observations 2

How define a good set of Colour and Depth parameters?

- According to the depth contrast of the image the parameters require to be adjusted:
  - Low Depth Contrast image $\Rightarrow$ larger values for colour parameters.

Observations 2

How define a good set of Colour and Depth parameters?

- According to the depth contrast of the image the parameters require to be adjusted:
  - Low Depth Contrast image $\Rightarrow$ larger values for colour parameters.
  - High Depth Contrast image $\Rightarrow$ larger values for depth parameters.

Observations 2

How define a good set of Colour and Depth parameters?

- According to the depth contrast of the image the parameters require to be adjusted:
  - Low Depth Contrast image $\Rightarrow$ larger values for colour parameters.
  - High Depth Contrast image $\Rightarrow$ larger values for depth parameters.
- Colour Smoothness term improves the results in zones of invalid depth pixels.

Observations 2

How define a good set of Colour and Depth parameters?

- According to the depth contrast of the image the parameters require to be adjusted:
  - Low Depth Contrast image $\Rightarrow$ larger values for colour parameters.
  - High Depth Contrast image $\Rightarrow$ larger values for depth parameters.
- Colour Smoothness term improves the results in zones of invalid depth pixels.
- Depth Smoothness term improves connectivity in the final result.

Observations 2

How define a good set of Colour and Depth parameters?

- According to the depth contrast of the image the parameters require to be adjusted:
  - Low Depth Contrast image $\Rightarrow$ larger values for colour parameters.
  - High Depth Contrast image $\Rightarrow$ larger values for depth parameters.
- Colour Smoothness term improves the results in zones of invalid depth pixels.
- Depth Smoothness term improves connectivity in the final result.

Remarks Second Approach

Remarks Second Approach

- The approach seems to be robust but results were not satisfactory.

Remarks Second Approach

- The approach seems to be robust but results were not satisfactory.
- There is still a lot of work to do in:

Remarks Second Approach

- The approach seems to be robust but results were not satisfactory.
- There is still a lot of work to do in:
  1. Define appropriate amount of Gaussian components for the FGDM and BGFDM.

Remarks Second Approach

- The approach seems to be robust but results were not satisfactory.
- There is still a lot of work to do in:
  1. Define appropriate amount of Gaussian components for the FGDM and BGFDM.
  2. Improve the seeding criteria.

Remarks Second Approach

- The approach seems to be robust but results were not satisfactory.
- There is still a lot of work to do in:
  1. Define appropriate amount of Gaussian components for the FGDM and BGFDM.
  2. Improve the seeding criteria.
  3. Adjusting the parameters involved in the energy function.

Third Approach: Robust Depth Based Seeding

- **Main Idea:** Implement appropriate criteria about depth data and object location to deduce good seeding.

Third Approach: Robust Depth Based Seeding

- **Main Idea:** Implement appropriate criteria about depth data and object location to deduce good seeding.
- In order to define the final seeds, functions W^1_{BG},W^2_{BG},W^3_{BG},and W^1_{FG},W^2_{FG},W^3_{FG} will be defined based in the following criteria:

Third Approach: Robust Depth Based Seeding

- **Main Idea:** Implement appropriate criteria about depth data and object location to deduce good seeding.
- In order to define the final seeds, functions W^1_{BG},W^2_{BG},W^3_{BG},and W^1_{FG},W^2_{FG},W^3_{FG} will be defined based in the following criteria:

Third Approach: Robust Depth Based Seeding

- **Main Idea:** Implement appropriate criteria about depth data and object location to deduce good seeding.
- In order to define the final seeds, functions W^1_{BG},W^2_{BG},W^3_{BG},and W^1_{FG},W^2_{FG},W^3_{FG} will be defined based in the following criteria:
  1. *Centre is FG, Border is BG.*

Third Approach: Robust Depth Based Seeding

- **Main Idea:** Implement appropriate criteria about depth data and object location to deduce good seeding.
- In order to define the final seeds, functions W^1_{BG},W^2_{BG},W^3_{BG},and W^1_{FG},W^2_{FG},W^3_{FG} will be defined based in the following criteria:
  1. *Centre is FG, Border is BG.*
  2. *Central and Closest is FG.*

Third Approach: Robust Depth Based Seeding

- **Main Idea:** Implement appropriate criteria about depth data and object location to deduce good seeding.
- In order to define the final seeds, functions W^1_{BG},W^2_{BG},W^3_{BG},and W^1_{FG},W^2_{FG},W^3_{FG} will be defined based in the following criteria:
  1. *Centre is FG, Border is BG.*
  2. *Central and Closest is FG.*
  3. *External and Planar is BG.*

Criteria 1: *Centre is FG, Border is BG*

Criteria 1: *Centre is FG, Border is BG*

1. Let (i, j) be the image coordinates of pixel p.

Criteria 1: *Centre is FG, Border is BG*

1. Let (i, j) be the image coordinates of pixel p.
2. Define $d_{centre}(p) = \max(\frac{|i-(r/2)|}{(r/2)}, \frac{|j-(c/2)}{(c/2)})$.

Criteria 1: *Centre is FG, Border is BG*

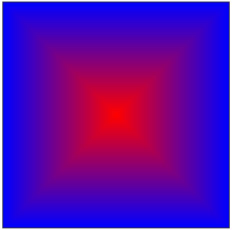
1. Let (i, j) be the image coordinates of pixel p.
2. Define $d_{centre}(p) = \max(\frac{|i-(r/2)|}{(r/2)}, \frac{|j-(c/2)}{(c/2)})$.
3. Set $W^1_{FG}(p) = 40 d_{centre}(p)$.

Criteria 1: *Centre is FG, Border is BG*

1. Let (i, j) be the image coordinates of pixel p.
2. Define $d_{centre}(p) = \max(\frac{|i-(r/2)|}{(r/2)}, \frac{|j-(c/2)}{(c/2)})$.
3. Set $W^1_{FG}(p) = 40 d_{centre}(p)$.
4. Set $W^1_{BG}(p) = 40(1 - d_{centre}(p))$.

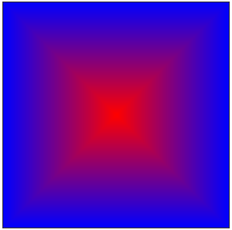
Criteria 1: *Centre is FG, Border is BG*

1. Let (i, j) be the image coordinates of pixel p.
2. Define $d_{centre}(p) = \max(\frac{|i-(r/2)|}{(r/2)}, \frac{|j-(c/2)|}{(c/2)})$.
3. Set $W^1_{FG}(p) = 40 d_{centre}(p)$.
4. Set $W^1_{BG}(p) = 40(1 - d_{centre}(p))$.



Criteria 1: *Centre is FG, Border is BG*

1. Let (i, j) be the image coordinates of pixel p.
2. Define $d_{centre}(p) = \max(\frac{|i-(r/2)|}{(r/2)}, \frac{|j-(c/2)|}{(c/2)})$.
3. Set $W^1_{FG}(p) = 40 d_{centre}(p)$.
4. Set $W^1_{BG}(p) = 40(1 - d_{centre}(p))$.



Criteria 2: *Central and Closest is Foreground*

Criteria 2: *Central and Closest is Foreground*

1. Use K-means to identify n clusters of depth data from the pixels in the central rectangle.

Criteria 2: *Central and Closest is Foreground*

1. Use K-means to identify n clusters of depth data from the pixels in the central rectangle.
2. Sort the clusters from closest to farthest. Let $d_{1,2}, d_{2,3}, \ldots, d_{n-1,n}$ be the distance between consecutive clusters.

Criteria 2: *Central and Closest is Foreground*

1. Use K-means to identify n clusters of depth data from the pixels in the central rectangle.
2. Sort the clusters from closest to farthest. Let $d_{1,2}, d_{2,3}, ..., d_{n-1,n}$ be the distance between consecutive clusters.
3. Label the closest cluster as FG.

Criteria 2: *Central and Closest is Foreground*

1. Use K-means to identify n clusters of depth data from the pixels in the central rectangle.
2. Sort the clusters from closest to farthest. Let $d_{1,2}, d_{2,3}, \ldots, d_{n-1,n}$ be the distance between consecutive clusters.
3. Label the closest cluster as FG.
4. If $d_{1,2} < d_{tol}$ second cluster is labelled FG.

Criteria 2: *Central and Closest is Foreground*

1. Use K-means to identify n clusters of depth data from the pixels in the central rectangle.
2. Sort the clusters from closest to farthest. Let $d_{1,2}, d_{2,3}, ..., d_{n-1,n}$ be the distance between consecutive clusters.
3. Label the closest cluster as FG.
4. If $d_{1,2} < d_{tol}$ second cluster is labelled FG.
5. If $d_{2,3} < m_{tol} d_{1,2}$ third cluster is labelled FG.

$$\vdots$$

Criteria 2: *Central and Closest is Foreground*

1. Use K-means to identify n clusters of depth data from the pixels in the central rectangle.
2. Sort the clusters from closest to farthest. Let $d_{1,2}, d_{2,3}, \ldots, d_{n-1,n}$ be the distance between consecutive clusters.
3. Label the closest cluster as FG.
4. If $d_{1,2} < d_{tol}$ second cluster is labelled FG.
5. If $d_{2,3} < m_{tol} d_{1,2}$ third cluster is labelled FG.

$$\vdots$$

6. If $d_{n-1,n} < m_{tol} d_{n-2,n-1}$ *nth* cluster is labelled FG

Criteria 2: *Central and Closest is Foreground*

1. Use K-means to identify n clusters of depth data from the pixels in the central rectangle.
2. Sort the clusters from closest to farthest. Let $d_{1,2}, d_{2,3}, ..., d_{n-1,n}$ be the distance between consecutive clusters.
3. Label the closest cluster as FG.
4. If $d_{1,2} < d_{tol}$ second cluster is labelled FG.
5. If $d_{2,3} < m_{tol} d_{1,2}$ third cluster is labelled FG.

$$\vdots$$

6. If $d_{n-1,n} < m_{tol} d_{n-2,n-1}$ *nth* cluster is labelled FG
7. From the pixels belonging to the clusters labelled FG, take the lowest value of depth **minDepthFG** and the largest value **maxDepthFG**.

Criteria 2: *Central and Closest is Foreground*

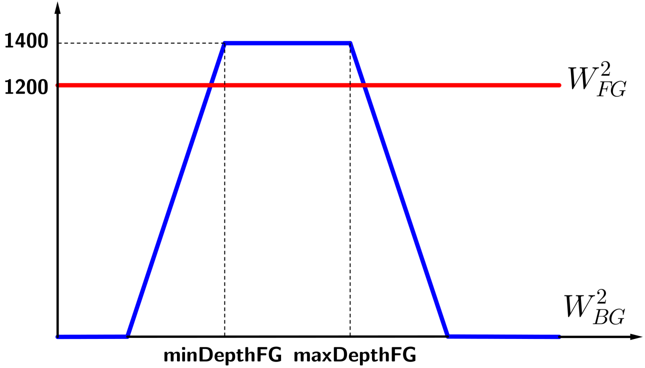
1. Use K-means to identify n clusters of depth data from the pixels in the central rectangle.
2. Sort the clusters from closest to farthest. Let $d_{1,2}, d_{2,3}, ..., d_{n-1,n}$ be the distance between consecutive clusters.
3. Label the closest cluster as FG.
4. If $d_{1,2} < d_{tol}$ second cluster is labelled FG.
5. If $d_{2,3} < m_{tol} d_{1,2}$ third cluster is labelled FG.

$$\vdots$$

6. If $d_{n-1,n} < m_{tol} d_{n-2,n-1}$ *nth* cluster is labelled FG
7. From the pixels belonging to the clusters labelled FG, take the lowest value of depth **minDepthFG** and the largest value **maxDepthFG**.

Criteria 2: *Central and Closest is Foreground*

8. For all p set $W^2_{FG}(p) = 1200$.
9. Set W^2_{BG} a stepwise linear function that separates between close pixels and far pixels respect to FG clusters.



Results: Central and Closest is Foreground

Results: Central and Closest is Foreground

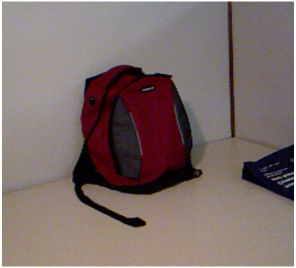
Results: Central and Closest is Foreground

Results: Central and Closest is Foreground

Results: Central and Closest is Foreground

Results: Central and Closest is Foreground

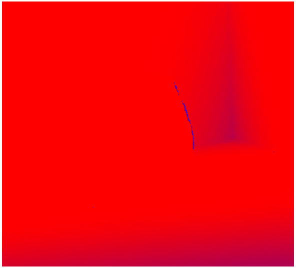




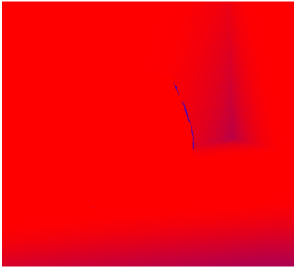
Results: Central and Closest is Foreground

Results: Central and Closest is Foreground



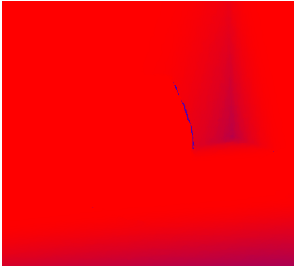



Results: Central and Closest is Foreground

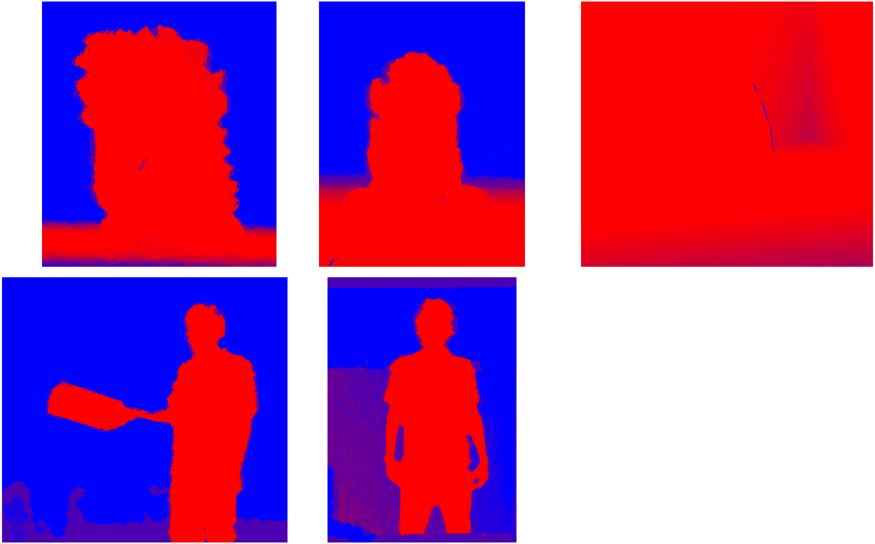


Results: Central and Closest is Foreground

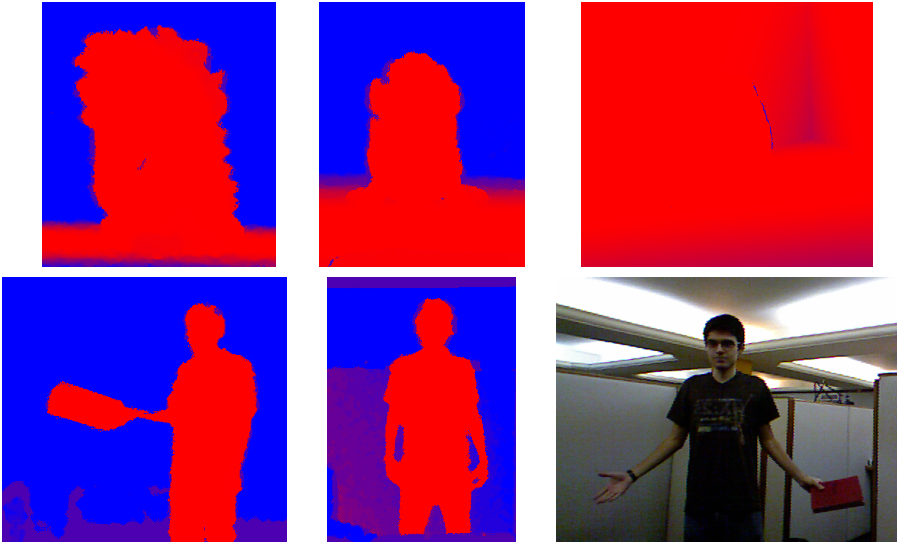




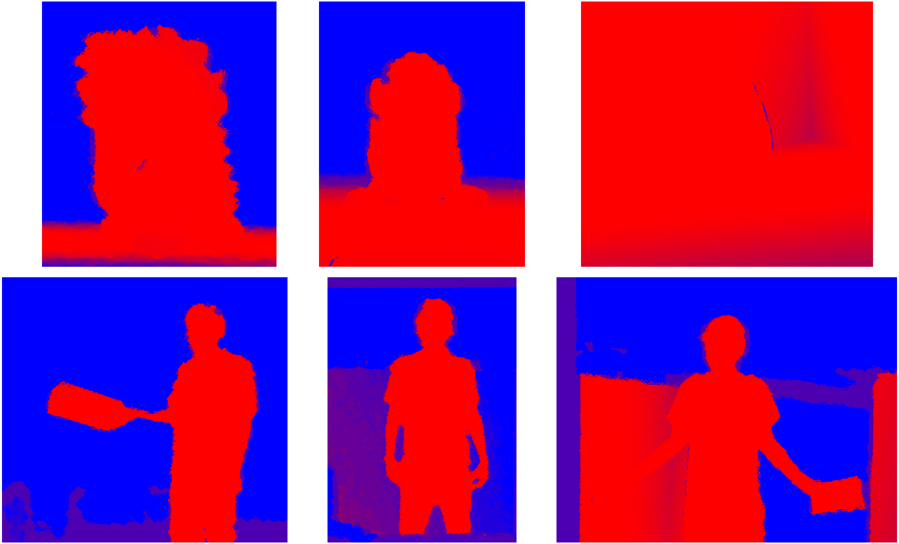
Results: Central and Closest is Foreground



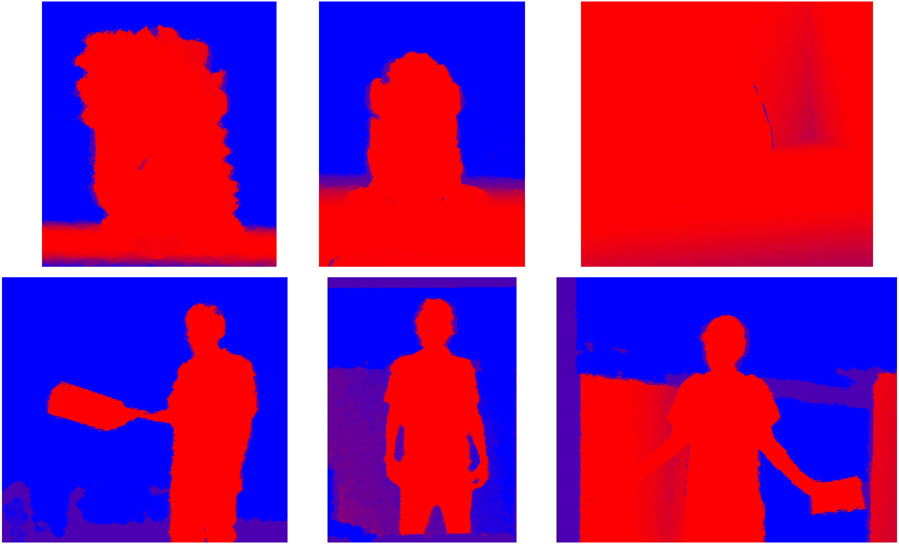
Results: Central and Closest is Foreground



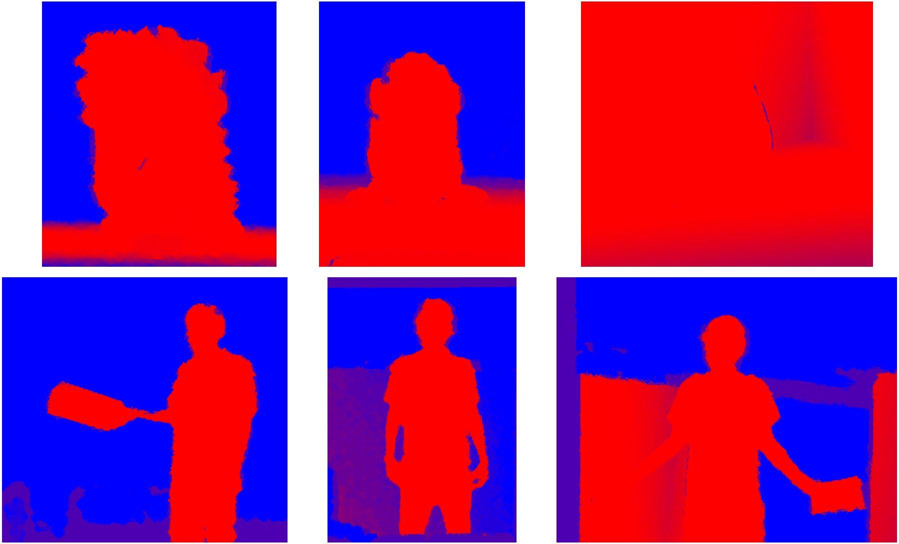
Results: Central and Closest is Foreground



Results: Central and Closest is Foreground



Results: Central and Closest is Foreground

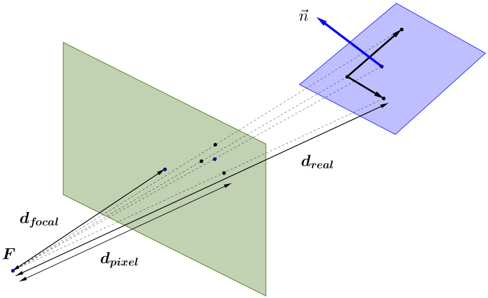


Criteria 3: *External and Planar is Background*

1. Calculate the normal at each pixel of the image.

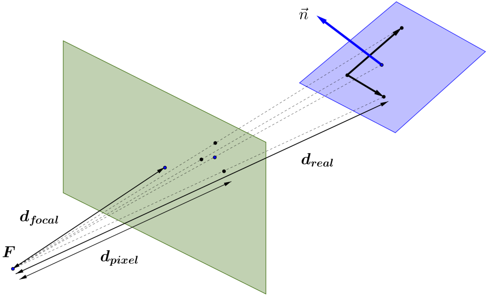
Criteria 3: *External and Planar is Background*

1. Calculate the normal at each pixel of the image.



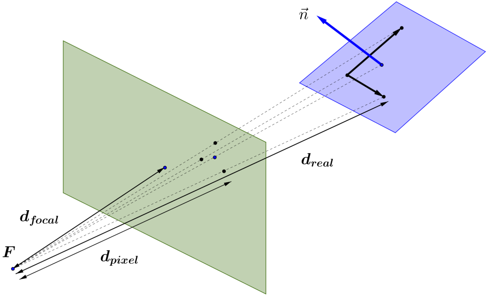
Criteria 3: *External and Planar is Background*

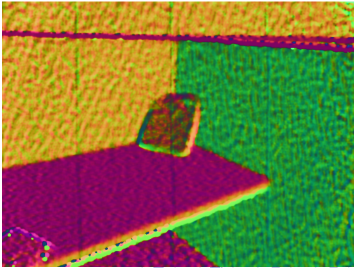
1. Calculate the normal at each pixel of the image.



Criteria 3: *External and Planar is Background*

1. Calculate the normal at each pixel of the image.



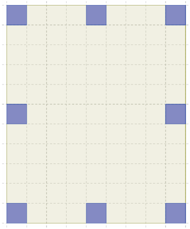


Criteria 3: *External and Planar is Background*

2. Take a sample of pixels from the border of the Selection Rectangle and store them in a queue.

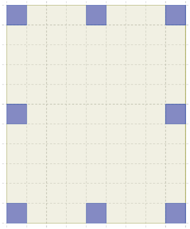
Criteria 3: *External and Planar is Background*

2. Take a sample of pixels from the border of the Selection Rectangle and store them in a queue.



Criteria 3: *External and Planar is Background*

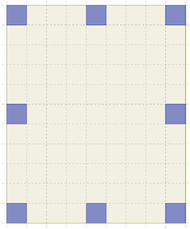
2. Take a sample of pixels from the border of the Selection Rectangle and store them in a queue.



3. Fix parameters for normal tolerance e_n and color tolerance e_c.

Criteria 3: *External and Planar is Background*

2. Take a sample of pixels from the border of the Selection Rectangle and store them in a queue.



3. Fix parameters for normal tolerance e_n and color tolerance e_c.
4. Pick the first pixel of the queue p and identify the connected set of pixels around p satisfying $||n_p - n_x|| < e_n$ and $||c_p - c_x|| < e_c$. Call this set of pixels $PN(p)$, and label all these pixels as *belong Component*.

Criteria 3: *External and Planar is Background*

5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:

Criteria 3: *External and Planar is Background*

5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:
    - $PN(p)$ is greater than 5% of image size.

Criteria 3: *External and Planar is Background*

5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:
   - $PN(p)$ is greater than 5% of image size.
   - At most 25% of $PN(p)$ belong to the central rectangle.

Criteria 3: *External and Planar is Background*

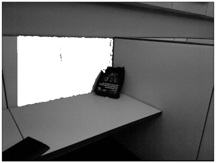
5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:
   - $PN(p)$ is greater than 5% of image size.
   - At most 25% of $PN(p)$ belong to the central rectangle.
6. If the previous conditions holds, we label all the pixels in PN(p) as *planar Pixels*. Otherwise, we undo the *belong Component* labelling, and we start to construct a new component from the next non *planar Pixel* in the queue.

Criteria 3: *External and Planar is Background*

5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:
   - $PN(p)$ is greater than 5% of image size.
   - At most 25% of $PN(p)$ belong to the central rectangle.
6. If the previous conditions holds, we label all the pixels in PN(p) as *planar Pixels*. Otherwise, we undo the *belong Component* labelling, and we start to construct a new component from the next non *planar Pixel* in the queue.

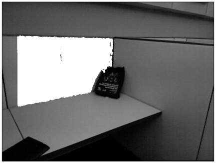
Criteria 3: *External and Planar is Background*

5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:
   - $PN(p)$ is greater than 5% of image size.
   - At most 25% of $PN(p)$ belong to the central rectangle.
6. If the previous conditions holds, we label all the pixels in PN(p) as *planar Pixels*. Otherwise, we undo the *belong Component* labelling, and we start to construct a new component from the next non *planar Pixel* in the queue.



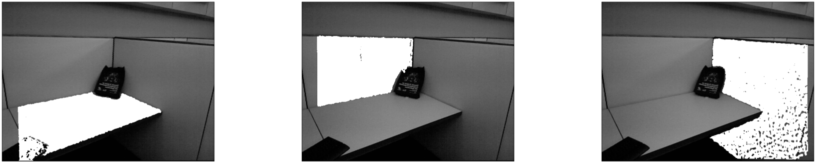
Criteria 3: *External and Planar is Background*

5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:
   - $PN(p)$ is greater than 5% of image size.
   - At most 25% of $PN(p)$ belong to the central rectangle.
6. If the previous conditions holds, we label all the pixels in PN(p) as *planar Pixels*. Otherwise, we undo the *belong Component* labelling, and we start to construct a new component from the next non *planar Pixel* in the queue.



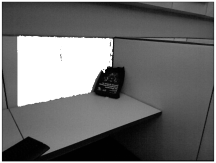
Criteria 3: *External and Planar is Background*

5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:
   - $PN(p)$ is greater than 5% of image size.
   - At most 25% of $PN(p)$ belong to the central rectangle.
6. If the previous conditions holds, we label all the pixels in PN(p) as *planar Pixels*. Otherwise, we undo the *belong Component* labelling, and we start to construct a new component from the next non *planar Pixel* in the queue.



Criteria 3: *External and Planar is Background*

5. In order to confirm $PN(p)$ as a valid plane the following two conditions must hold:
   - $PN(p)$ is greater than 5% of image size.
   - At most 25% of $PN(p)$ belong to the central rectangle.
6. If the previous conditions holds, we label all the pixels in PN(p) as *planar Pixels*. Otherwise, we undo the *belong Component* labelling, and we start to construct a new component from the next non *planar Pixel* in the queue.

7. Set $W^3_{BG}(p) = 0$ for all pixels p. Set $W^3_{FG}(p) = 1200 \iff p$ is *planar Pixel*.

Results: External and Planar is Background

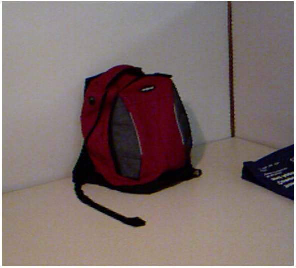
Results: External and Planar is Background

Results: External and Planar is Background

Results: External and Planar is Background

Results: External and Planar is Background

Results: External and Planar is Background

Results: External and Planar is Background



Results: External and Planar is Background

Results: External and Planar is Background

Results: External and Planar is Background

Results: External and Planar is Background





Results: External and Planar is Background

Results: External and Planar is Background

Results: External and Planar is Background

Seeding

Seeding

1. Let $W_{FG} := W^1_{FG} + W^2_{FG} + W^3_{FG}$ and $W_{BG} := W^1_{BG} + W^2_{BG} + W^3_{BG}$.

Seeding

1. Let $W_{FG} := W_{FG}^1 + W_{FG}^2 + W_{FG}^3$ and $W_{BG} := W_{BG}^1 + W_{BG}^2 + W_{BG}^3$.
2. If $W_{BG}(p) - W_{FG}(p) > 160$, then p is a **FG seed**.

Seeding

1. Let $W_{FG} := W_{FG}^1 + W_{FG}^2 + W_{FG}^3$ and $W_{BG} := W_{BG}^1 + W_{BG}^2 + W_{BG}^3$.
2. If $W_{BG}(p) - W_{FG}(p) > 160$, then p is a **FG seed**.
3. If $W_{FG}(p) - W_{BG}(p) > 450$, then p is a **BG seed**.

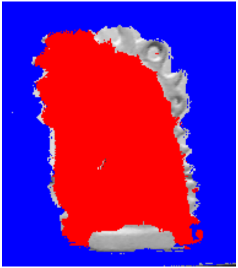
Seeding

1. Let $W_{FG} := W^1_{FG} + W^2_{FG} + W^3_{FG}$ and $W_{BG} := W^1_{BG} + W^2_{BG} + W^3_{BG}$.
2. If $W_{BG}(p) - W_{FG}(p) > 160$, then p is a **FG seed**.
3. If $W_{FG}(p) - W_{BG}(p) > 450$, then p is a **BG seed**.

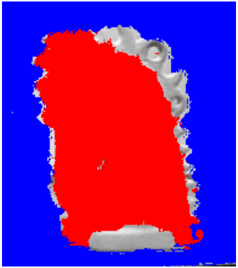
Results: Seeding

Results: Seeding

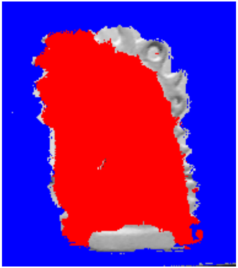
Results: Seeding

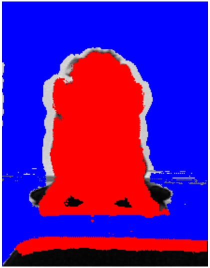


Results: Seeding

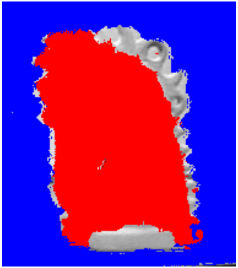
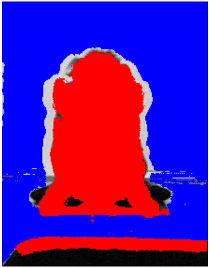
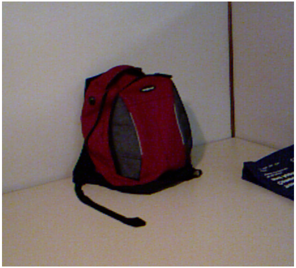


Results: Seeding

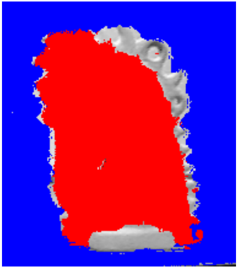


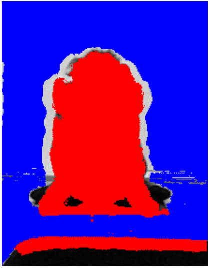


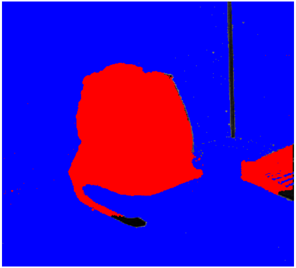
Results: Seeding

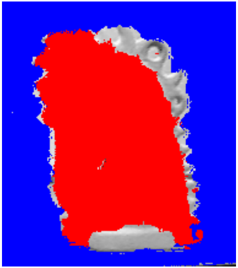
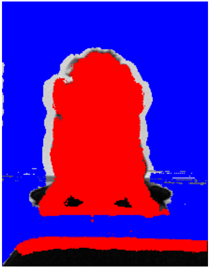
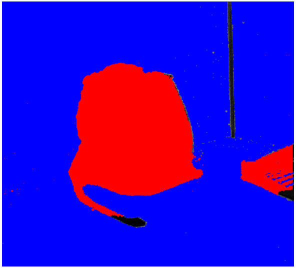
Results: Seeding



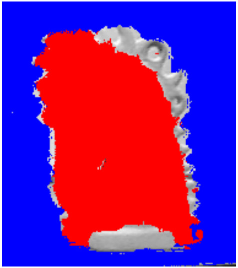


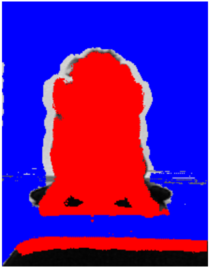


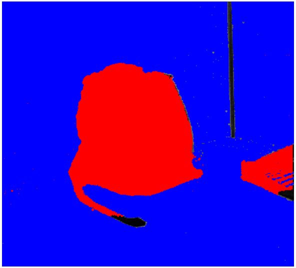
Results: Seeding

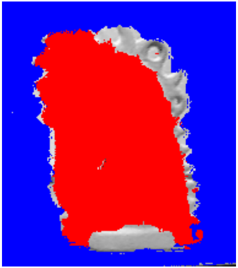
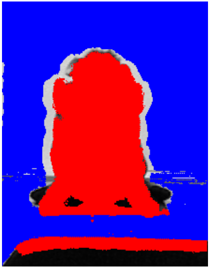
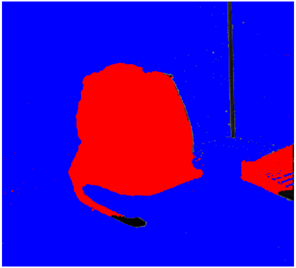

Results: Seeding



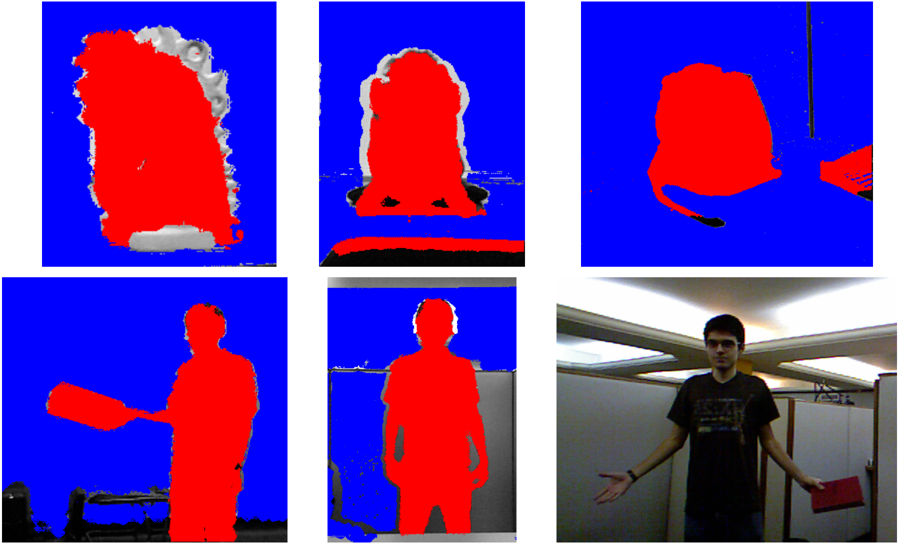




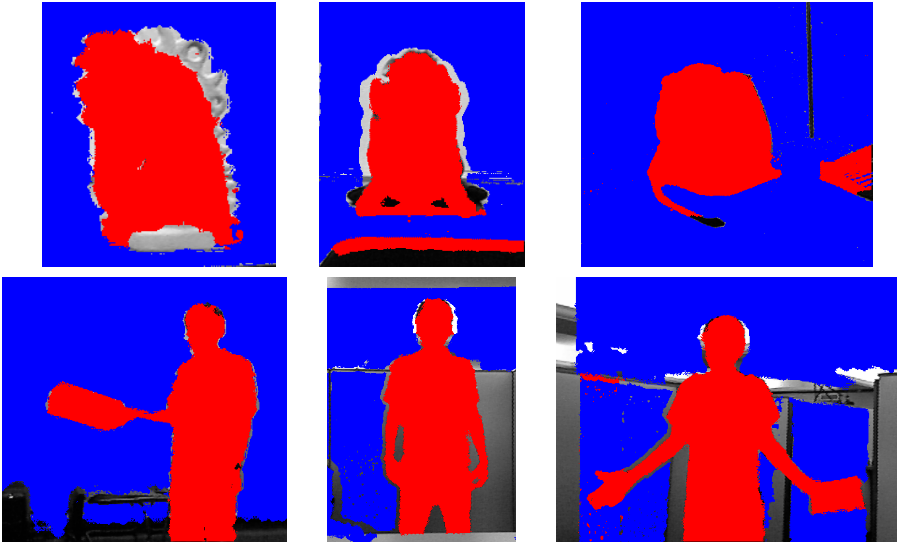
Results: Seeding

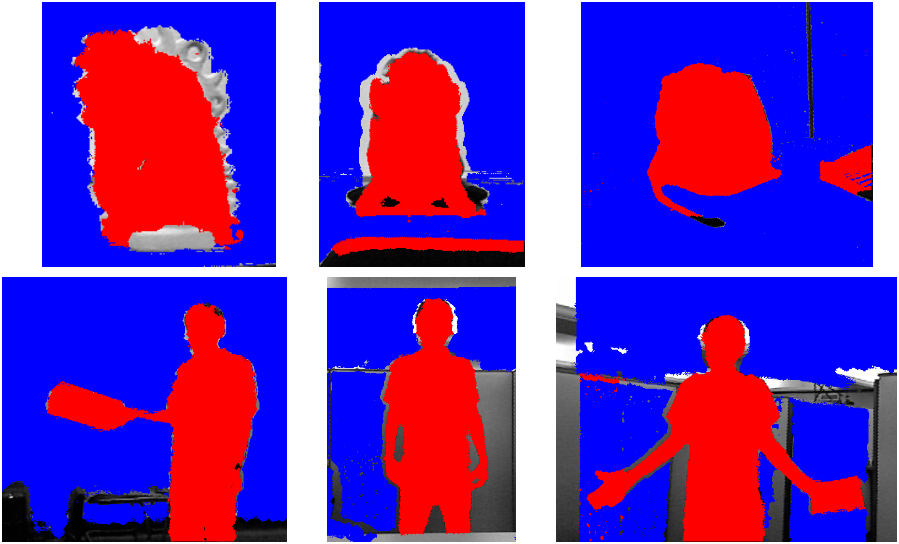
Results: Seeding



Results: Seeding



Results: Seeding



Energy Function

$$E(x, \omega_C, z, d) = U(x, \omega_C, z, d) + V(x, z, d)$$

Energy Function

$$E(x, \omega_C, z, d) = U(x, \omega_C, z, d) + V(x, z, d)$$

$$\begin{aligned} U(x, \omega_C, z, d) =& U_C(x, \omega_C, z) \rightarrow \text{As GrabCut} \\ &+ \alpha_D \Big(\sum_p W_{BG}(d_p)[x_p = 0] + W_{FG}(d_p)[x_p = 1] \Big). \end{aligned}$$

Energy Function

$$E(x, \omega_C, z, d) = U(x, \omega_C, z, d) + V(x, z, d)$$

$$\begin{aligned} U(x, \omega_C, z, d) =& U_C(x, \omega_C, z) \rightarrow \text{As GrabCut} \\ &+ \alpha_D \Big(\sum_p W_{BG}(d_p)[x_p = 0] + W_{FG}(d_p)[x_p = 1] \Big). \end{aligned}$$

$$\begin{aligned} V(x, z, d) =& V_C(x, z) \rightarrow \text{As GrabCut} \\ &+ \gamma_D \Big(\sum_{(p,q \in N)} (1 - \frac{(d_p - d_q)^2}{600 + (d_p - d_q)^2})[x_p \neq x_q] \Big). \end{aligned}$$

Results:

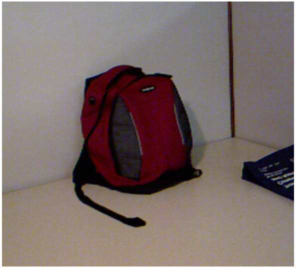
Results:

Results:

Results:

Results:

Results:



Results:



Results:

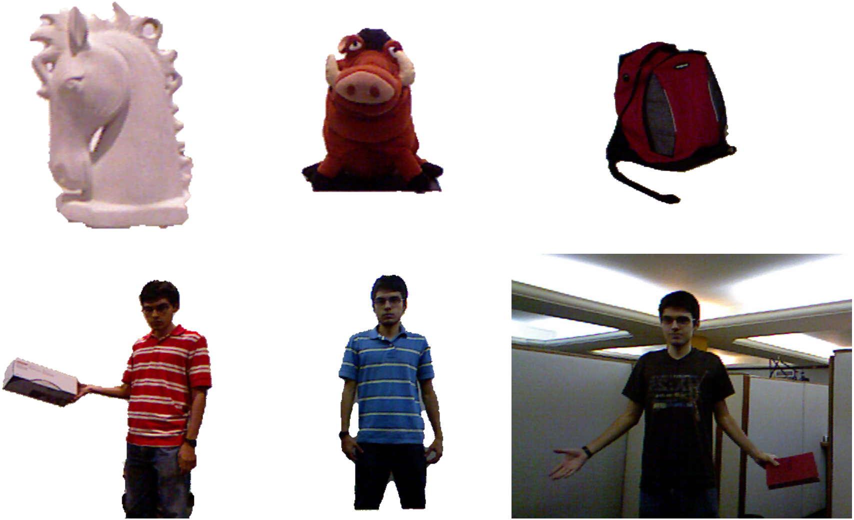



Results:

Results:

Results:

Results:



Results:

Results:

Remarks Third Approach

Remarks Third Approach

- Pros:

Remarks Third Approach

- Pros:
  1. Works in object-background adjacency and non adjacency.

Remarks Third Approach

- Pros:
  1. Works in object-background adjacency and non adjacency.
  2. Stronger to depth noise.

Remarks Third Approach

- Pros:
  1. Works in object-background adjacency and non adjacency.
  2. Stronger to depth noise.
- Cons:

Remarks Third Approach

- Pros:
  1. Works in object-background adjacency and non adjacency.
  2. Stronger to depth noise.
- Cons:
  1. Parameters still requires calibration for specific cases.

Remarks Third Approach

- Pros:
    1. Works in object-background adjacency and non adjacency.
    2. Stronger to depth noise.
- Cons:
    1. Parameters still requires calibration for specific cases.
    2. Seeding still unstable in the clustering step.

Remarks Third Approach

- Pros:
  1. Works in object-background adjacency and non adjacency.
  2. Stronger to depth noise.
- Cons:
  1. Parameters still requires calibration for specific cases.
  2. Seeding still unstable in the clustering step.

Conclusions

Conclusions

- First Approach propose a simple and effective method for object extraction in the case of object-background non adjacency. It must improve in depth noise manipulation.

Conclusions

- First Approach propose a simple and effective method for object extraction in the case of object-background non adjacency. It must improve in depth noise manipulation.
- Second Approach looks for a robust segmentation including data depth in the probabilistic model of GrabCut, but we didn't get an accurate representation of depth distribution from Gaussian Mixture Model

Conclusions

- First Approach propose a simple and effective method for object extraction in the case of object-background non adjacency. It must improve in depth noise manipulation.
- Second Approach looks for a robust segmentation including data depth in the probabilistic model of GrabCut, but we didn't get an accurate representation of depth distribution from Gaussian Mixture Model
- Third Approach improves the seeding using planar subtraction and gets good results in general cases. It still requires calibration.

Further Research: Video Object Extraction

1. Extract the object from the initial frame and define accurate FG and BG Colour Models.(Use the Third Approach).

Further Research: Video Object Extraction

1. Extract the object from the initial frame and define accurate FG and BG Colour Models.(Use the Third Approach).
2. Construct a contour graph around segmentation result.

Further Research: Video Object Extraction

1. Extract the object from the initial frame and define accurate FG and BG Colour Models.(Use the Third Approach).
2. Construct a contour graph around segmentation result.

Further Research: Video Object Extraction

1. Extract the object from the initial frame and define accurate FG and BG Colour Models.(Use the Third Approach).
2. Construct a contour graph around segmentation result.

3. Apply GrabCut in the next frame just in the pixels belonging to the contour graph.

Further Research: Video Object Extraction

1. Extract the object from the initial frame and define accurate FG and BG Colour Models.(Use the Third Approach).
2. Construct a contour graph around segmentation result.


3. Apply GrabCut in the next frame just in the pixels belonging to the contour graph.
4. Update the contour graph from the new segmentation result

Further Research: Video Object Extraction

1. Extract the object from the initial frame and define accurate FG and BG Colour Models.(Use the Third Approach).
2. Construct a contour graph around segmentation result.

3. Apply GrabCut in the next frame just in the pixels belonging to the contour graph.
4. Update the contour graph from the new segmentation result

Further Research: Video Object Extraction

1. Extract the object from the initial frame and define accurate FG and BG Colour Models.(Use the Third Approach).
2. Construct a contour graph around segmentation result.


3. Apply GrabCut in the next frame just in the pixels belonging to the contour graph.
4. Update the contour graph from the new segmentation result


Further Research: Video Object Extraction

1. Extract the object from the initial frame and define accurate FG and BG Colour Models.(Use the Third Approach).
2. Construct a contour graph around segmentation result.


3. Apply GrabCut in the next frame just in the pixels belonging to the contour graph.
4. Update the contour graph from the new segmentation result


5. Return to step 3.

Further Research: Video Object Extraction

The previous result was obtained using just colour data in the energy function.

Further Research: Video Object Extraction

The previous result was obtained using just colour data in the energy function.

Why not include depth data and optical flow data in the energy function?...

Thanks to Djalma, Francisco, Leandro, Lucas, and professor Luiz, for suggestions and support!.

Y. Boykov and M-P. Jolly. *Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images*. Proc. IEEE Int. Conf. on Computer Vision, 2001.

C. Rother, V. Kolmogorov, and A. Blake. *Grabcut interactive foreground extraction using iterative graph cuts*. Proc. ACM Siggraph, 2004.

Y. Boykov and V. Kolmogrov. *An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision*. IEEE Trans. on Pattern Analysis and Machine Intelligence, volume 26, pages 11241137, 2004.

C. Rother, V. Kolmogorov, Y. Boykov, and A. Blake. *Interactive Foreground Extraction using graph cut*. Microsoft Technical Report:MSR-TR-2011-46.