

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
КАФЕДРА ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

«До захисту допущено»

Завідувач кафедри

(підпис)
М. В. Грайворонський

“ ”
(ініціали, прізвище)

2017 р.

Дипломна робота

освітньо-кваліфікаційного рівня “магістр”

за спеціальністю 8.04030101 «Прикладна математика»

на тему «Тема»

Виконав студент 6 курсу групи group

Name

Керівник Rank, Name

Рецензент Rank, Name

(підпис)

(підпис)

(підпис)

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авто-
рів без відповідних посилань.

Студент _____

ЗМІСТ

Вступ	6
0.1 Предисловие	6
0.2 Способы восприятия глубины	6
1 Теоретичні відомості	9
1.1 Нахождение абсолютного расстояния методом бинокулярного параллакса	9
1.2 Стереозрение при полных данных	13
1.3 Стерезрение при неполных данных.	17
1.4 Предвычисления на неполных данных	17
2 Практичні результати	18
3 final remarks.	19
Висновки	20
Перелік посилань	22

ВСТУП

0.1 Предисловие

Нахождение карты глубины сцены является одной из важнейших задач в области компьютерного зрения. Решение этой задачи может осуществляться несколькими способами. Хотя они все и разные, но принцип у всех один и тот же – чем дальше объект, тем меньше "изменения".

Глаз = любой зрительный сенсор (человеческий глаз/ камера и т.п)

0.2 Способы восприятия глубины

Восприятие глубины (восприятие расстояния) — зрительная способность воспринимать действительность в её трёх измерениях, воспринимать расстояние до объекта.

Данное восприятие формируется при помощи множества так называемых "глубинных признаков". Это такие характеристики, которые у одного и того же объекта на разных расстояниях – различны. Их можно разделить на Монокулярные – те, для обнаружения которых будет достаточно и одного глаза / сенсора, и Бинокулярные – полноценное восприятие которых возможно только как агрегация сенсорной информации поступающей с двух глаз. Так же, иногда их делят на Динамические – требующие движения глаза или самого объекта, и Стационарные – для которых подобные действия не нужны.

0.2.1 Монокулярные глубинные признаки

- Параллакс

Один из самых знакомых нам признаков, и очень широко используемый эффект в задачах нахождения карты глубины. Суть его проста – чем дальше объект, тем меньше он будет смещаться (на проекции) при движении глаза.

.Parallax.png

- optical expansion!! перевести

Если объект движется на/от нас, размер его проекции будет увеличиваться/ уменьшаться.

.Optical_expansion.png

- Относительные размеры

Если известно, что несколько объектов имеют похожие размеры (напр. люди), причём абсолютные размеры могут быть неизвестны, то на основании размеров их проекций можно сделать выводы про относительное расстояние до них (Если один человек на снимке значительно больше остальных, значит, скорее-всего, оно находится ближе всего к камере).

.Relative_sizes.png

- Градиент текстуры

Хорошо различимые мелкие детали текстуры поверхности становятся всё хуже заметными с расстоянием.

.Texture_gradient.png

- Свет и тени

Тени отбрасываемые объектами, то как свет падает и отражается от объектов, позволяют определять форму объектов и их положение в пространстве.

.Lightshadows.pngsiluets, contours, shadows

0.2.2 Бинокулярные глубинные признаки

- Бинокулярный параллакс

При использовании двух изображений одной и той же сцены, полученных под разными углами, можно триангулировать расстояние до объекта с высокой степенью точности. Именно на этом принципе основано 3D-кино и автостереограммы.

.Binocularparallax.png

- Конвергенция

Присущий живым организмам признак, конвергенция – кинестические ощущения от сокращения глазных мышц при фокусировке на каком-либо объекте.

.Convergence.png

Это далеко не все существующие глубинные признаки. Более полный список можно найти здесь: [TeX\[1\]](#)

В данной работе будет рассматриваться использование только бинокулярного параллакса. И хоть этот метод сам по себе не даёт абсолютной информации о глубине сцены, зная некоторые дополнительные параметры её можно получить.

1 ТЕОРЕТИЧНІ ВІДОМОСТІ

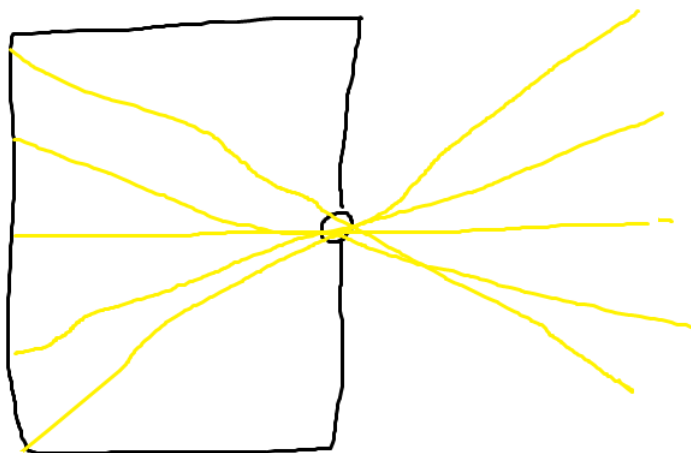
1.1 Нахождение абсолютного расстояния методом бинокулярного параллакса

1.1.1 О камерах

В нашем случае в качестве ”глаза” используется камера, поэтому для последующих выкладок нам нужно иметь представление о том, как она устроена и как работает.

Сегодня существует множество разных типов камер, и хоть каждая из них может иметь свои особенности конструкции, базовый принцип работы у них у всех одинаковый.

Основа камеры – её светочувствительная матрица. В ней свет преобразуется в поток цифровых данных, непосредственно формируя изображение. После неё обычно находится объектив. Он отвечает за проекцию изображения на матрицу, позволяя регулировать фокусное расстояние, значение диафрагмы и другие параметры.

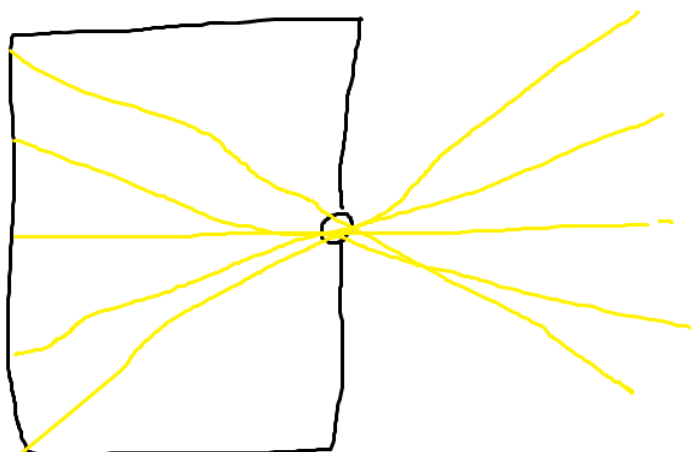


В рамках этой работы, нам будет достаточно рассмотреть камеру самой

простой конструкции. *Pinholecamera* или же

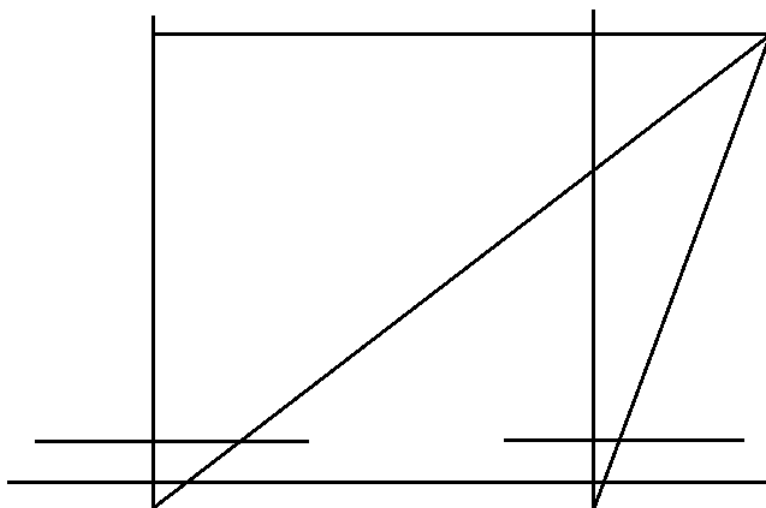
!!!!!!

– одна из самых первых "камер". В такой камере отсутствует линза, вместо неё выступает маленькое отверстие в передней части камеры.



В силу оптических особенностей, конечное изображение в такой камере будет перевернутым. Поэтому для ещё большего упрощения мысленно перенесём матрицу камеры направо, на расстояние f . Такое действие никак не повлияет на итоговый результат, но сильно упростит дальнейшие выкладки.

Покажем теперь, как с помощью двух таких камер можно найти абсолютное расстояние до объекта. Пусть пока камеры будут расположены параллельно и сонаправлены друг другу (что делать в случае когда камеры расположены иначе будет указано дальше). Итак, у нас есть такая конструкция:



Где f – фокусное расстояние камеры, Δ – расстояние между камерами, z – искомая глубина изображения, x_1, x_2 – координата объекта на левом и на правом снимке, R, L – оптические центры правой и левой камеры, S – объект.

Тогда из подобия треугольников следует:

$$\begin{cases} \frac{f}{x_1} = \frac{z}{\Delta + x} \\ \frac{f}{x_2} = \frac{z}{x} \end{cases}$$

Выразив Δ из первого уравнения, и x из второго получим:

$$\begin{cases} \Delta = \frac{zx_1}{f} - x \\ x = \frac{zx_2}{f} \end{cases}$$

Подставим x в первое уравнение и выразим из получившегося z

$$z = \frac{\Delta f}{(x_1 - x_2)}$$

Так как Δ и f не зависят от расположения объекта, а являются характеристиками самой камеры, то расстояние до объекта зависит только от разницы $x_1 - x_2$, то есть только от смещения объекта между двумя снимками. Следовательно для нахождения карты глубин нам нужно найти "сдвиг" для каждого

пикселя.

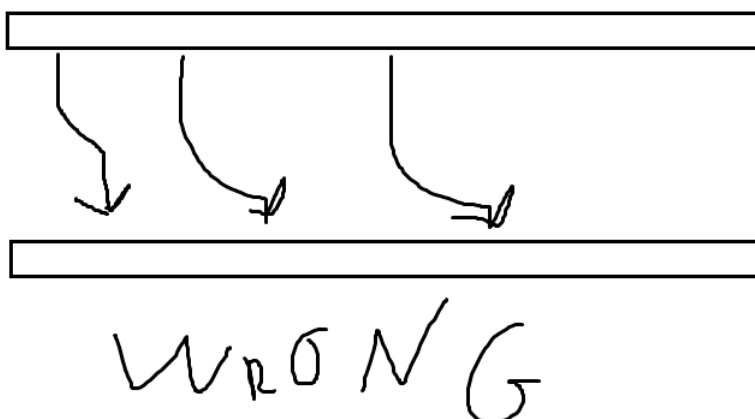
1.1.2 !!Ректификация!!

Но для того чтобы вычислить $x_1 - x_2$ нужно сначала найти эту пару соответствующих пикселей, а это сложная задача. Для каждого пикселя левого изображения, нам нужно найти соответствующий среди пикселей правого изображения. Нужно ли нам перебирать все пиксели правого изображения, или всё таки можно ограничиться каким-то подмножеством пикселей?

Пусть у нас есть изображение какого-то объекта на фоне стены.

6

Всё что мы можем сказать о его расположении относительно камеры, это что он находится где-то на луче L .



Добавим теперь вторую камеру, на каком-то расстоянии Δ от первой камеры. Если мы теперь проведем лучи из оптического центра правой камеры

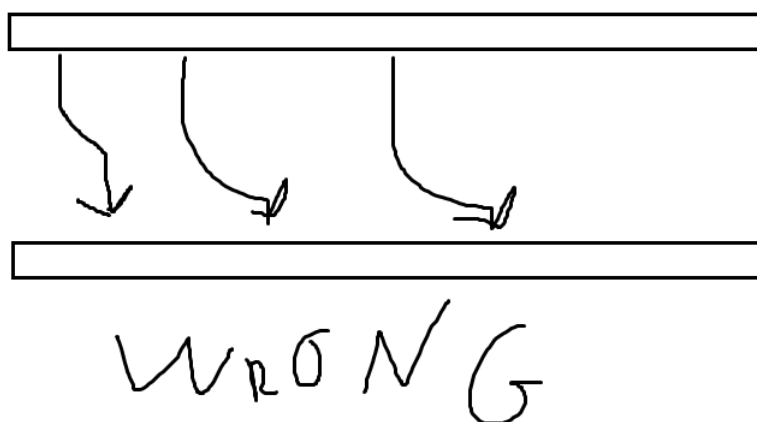
так, чтобы они пересекали луч L - мы получим, так называемые, "эпиполярные линии". И теперь мы можем сократить множество для поиска соответствующих пар со всего изображения до эпиполярных линий на нём. Это существенно облегчает нам задачу.

1.2 Стереозрение при полных данных

тест тест тест тест тест тест тест тест тест тест тест тест тест тест тест
тест тест тест тест тест тест тест тест тест тест тест тест тест тест тест тест
тест тест тест тест тест тест тест тест тест тест тест тест тест тест тест тест
тест тест тест тест тест тест тест тест тест тест тест тест тест тест тест

1.2.1 Одномерный случай

Для простоты рассмотрим алгоритм решения задачи стереозрения в случае когда наши изображения представляют собой строки пикселей. Нам нужно каждому пикселю из левой строки поставить в соответствие пиксель из правой:



Однако не любая пара пикселей может находиться в соответствии! Пикселю левого изображения с номером i могут соответствовать только $i - k$ -е

пиксели правого изображения ($k = (0, i - 1)$). Вы можете убедиться в этом держа перед собой карандаш и поочередно закрывая то правый, то левый глаз. В правом глазу карандаш будет находиться левее чем в левом.

9

Но всё же, как нам искать соответствующие пиксели? Какими характеристиками обладают соответствующие пиксели? Логично было бы предположить, что у них одинаковый цвет. Но этого не достаточно. Тогда можно ещё посмотреть какие свойства есть у соседних соответствующих пикселей. Допустим для пикселя x_i уже найден соответствующий ему пиксель, то есть нам уже известне сдвиг для (i) -го пикселя. Можем ли мы что-то сказать о сдвиге для $(i + 1)$ -го пикселя? Так как всё же большинство объектов в реальной жизни - гладкие, то велика вероятность, что сдвиг для $(i + 1)$ -го пикселя не будет сильно отличаться от сдвига (i) -го пикселя. Иными словами в последовательности сдвигов должно быть мало скачков.

Итак, у нас есть два признака похожести пикселей: схожесть цветов и схожесть соседних сдвигов. Введём качественные оценки этих двух характеристик. Схожесть цветов определим как расстояние между ними, в их цветовом пространстве (!опасный момент!). Схожесть же соседних сдвигов определим просто как модуль разности значений этих сдвигов.

Поставим задачу более формально. Пусть множество $\mathcal{L} = \{x_i \mid i = \overline{1, n}\}$ — левое изображение-строка, n - ширина изображения, x_i - цвет пикселя (если у нас черно-белые 8-битные изображения, то $x_i \in [0, 255]$, если же цветные, то $x_i \in [0, 255]^3$). Введём функцию $\mathcal{L}(i)$ — цвет i -го пикселя на левом изображении. Аналогично, \mathcal{R} — правое изображение-строка, а $\mathcal{R}(i)$ —

цвет i -го пикселя на правом изображении.

Для каждого пикселя i левого изображения нам нужно найти соответствующий ему пиксель на правом изображении, то есть найти такой сдвиг d_i , что:

1) Минимизирует **Унарный штраф** $H(i, d) = | \mathcal{L}(i) - \mathcal{R}(i - d) |$
(!норма, а не модуль!)

2) Минимизирует **Бинарный штраф** $g(d, d') = \alpha | d - d' |$

(где α — коэффициент сглаживания, и подбирается экспериментально)

Можем построить штрафную функцию $\omega(\bar{d}) = \sum_{i=1}^n H(i, d_i) + \sum_{i=1}^{n-1} g(d_i, d_{i+1})$

Тогда, такая последовательность \bar{d} которая минимизирует штрафную функцию $\omega(\bar{d})$ и будет нашим решением. Таким образом имеем задачу оптимизации.

Для нахождения эффективного решения, и большего понимания задачи, представим её в несколько ином виде, а именно в виде ориентированного графа.

Множество вершин $V = \{ \sigma(i, d) \mid i = \overline{(1, n)}, d = \overline{(0, \max D)} \} \cup \{ S, E \}$.

Вес вершины $\sigma(i, d) = H(i, d)$, $i = \overline{(1, n)}$, $d = \overline{(0, \max D)}$

Веса ребёр:

1) Из S в $\sigma(1, d)$ вес ребра = 0, $d = \overline{(0, \max D)}$

2) Из $\sigma(n, d)$ в E вес ребра = 0, $d = \overline{(0, \max D)}$

3) Из $\sigma(i, d)$ в $\sigma(i + 1, d')$ вес ребра = $g(d, d')$, $d, d' = \overline{(0, \max D)}$

Получается такой граф:

10

Тогда, длина пути из вершины S в вершину E через вершины $\sigma(i, d)$ где $i = \overline{(1, n)}$, а $d \in \bar{d}$ будет равна:

$$\sum_{i=1}^n H(i, d_i) + \sum_{i=1}^{n-1} g(d_i, d_{i+1})$$

Что в точности повторяет нашу штрафную функцию, причём последовательность \bar{d} минимизирующая эту штрафную функцию, будет отвечать последовательности вершин, которые составляют самый короткий путь из вершины S в вершину E .

Таким образом наша задача сводится к поиску кратчайшего пути на графе.

Решать эту задачу традиционными алгоритмами типа Белмана-Форда или Дейкстры - не самая лучшая идея. Алгоритм Белмана-Форда имеет сложность $|V| * |E|$, причём $|E|$ у нас равна D^2n , где D - максимальный дисперситет, n - ширина изображения.

Куда эффективнее решать эту задачу при помощи ДИНАМИЧЕСКОГО ПРОГРАМИРОВАНИЯ.

Обозначим длину кратчайшего пути из вершины S в вершину $\sigma(i, d)$ как $f_i(d)$. Тогда для любого $d \in D$:

$$f_1(d) = H(1, d)$$

$$f_2(d) = \min(H(1, d') + g(d', d)) = \min(f_1(d') + g(d', d)) + H(2, d)$$

...

$$f_n(d) = \min(f_{n-1}(d') + g(d', d)) + H(n, d)$$

А саму последовательность \bar{d} находим так:

$$d_m = \arg \min_{d'} (f_m(d'))$$

$$d_i = \arg \min_{d'} (f_i(d') + H(i, d'))$$

2 ПРАКТИЧНІ РЕЗУЛЬТАТИ

3 FINAL REMARKS

ВИСНОВКИ

В результаті виконання роботи вдалося.

REFERENCES

ПЕРЕЛІК ПОСИЛАНЬ

- 1 wikipedia. The T_EXbook / wikipedia // Computers & typesetting. — test, 1984.