**Масштабирование изображений в Matlab**

Для масштабирования изображения в Matlab используется функция imresize. В качестве аргументов функции можно указывать:

* изображение для масштабирования I,
* фактор масштабирования scale, представленный действительным числом**:**

I2 = imresize(I,0.5);

* целевой размер изображения outputSize, заданный массивом [numrows numcols], где numrows или numcols может иметь значение NaN:

I2 = imresize(I,[50 NaN]);

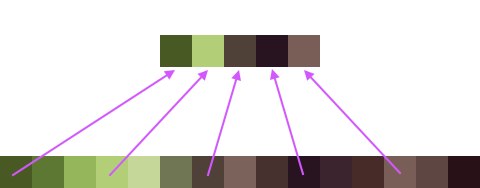
* Метод масштабирования method, заданный строкой:

I2 = imresize(I,0.5,'bilinear');

**Поддерживаемые методы масштабирования изображения:**

**Ближайший сосед (Nearest neighbor)**

Метод ближайшего соседа задается параметром 'nearest'. Это самый примитивный и быстрый метод интерполяции. Для каждого пикселя конечного изображения выбирается один пиксель исходного, наиболее близкий к его положению с учетом масштабирования. Такой метод дает пикселизированное изображение при увеличении и **сильно зернистое изображение при уменьшении**.



Качество и производительность любого метода уменьшения можно оценить по отношению количества пикселей, участвовавших в формировании конечного изображения, к числу пикселей в исходном изображении. Чем больше это отношение, тем алгоритм качественнее и медленнее. Отношение, равное одному, означает что как минимум каждый пиксель исходного изображения сделал свой вклад в конечное.

Например, если мы уменьшаем изображение методом ближайшего соседа в 3 раза по каждой стороне, то это соотношение равно 1/9, то есть большая часть исходных пикселей никак не учитывается. Метод ближайшего соседа применяется для уменьшения крайне редко, так как дает очень плохое качество, хотя и может быть полезен при увеличении.

**Аффинные преобразования (Affine transformations)**

Аффинные преобразования – общий метод для искажения изображений. Они позволяют за одну операцию повернуть, растянуть и отразить изображение. Поэтому во многих приложениях и библиотеках, реализующих метод аффинных преобразований, функция изменения изображений является просто оберткой, рассчитывающей коэффициенты для преобразования.

Выделяют методы билинейной интерполяции (задается параметром 'bilinear') и бикубической интерполяции (задается параметром 'bicubic' и устанавливается по умолчанию).

Принцип действия заключается в том, что для каждой точки конечного изображения берется фиксированный набор точек исходного и интерполируется в соответствии с их взаимным положением и выбранным фильтром. Количество точек тоже зависит от фильтра.

Для **билинейной интерполяции** берется 2x2 исходных пикселя, для **бикубической** 4x4. Такой метод дает гладкое изображение при увеличении, **но при уменьшении результат очень похож на ближайшего соседа**.

Теоретически, при бикубическом фильтре и уменьшении в 3 раза отношение обработанных пикселей к исходным равно 4² / 3² = 1,78. На практике результат значительно хуже так как в существующих реализациях окно фильтра и функция интерполяции не масштабируются в соответствии с масштабом изображения, и пиксели ближе к краю окна берутся с отрицательными коэффициентами (в соответствии с функцией), то есть не вносят полезный вклад в конечное изображение. В результате изображение, уменьшенное с бикубическим фильтром, отличается от изображения, уменьшенного с билинейным, только тем, что оно еще более четкое.

Для билинейного фильтра и уменьшения в три раза отношение обработанных пикселей к исходным равно 2² / 3² = 0.44, что принципиально не отличается от ближайшего соседа. Фактически, аффинные преобразования нельзя использовать для уменьшения более чем в 2 раза. И даже при уменьшении до двух раз они дают заметные эффекты лесенки для линий.

Время работы заметно больше, чем у ближайшего соседа, и зависит от размера конечного изображения и размера окна выбранного фильтра.

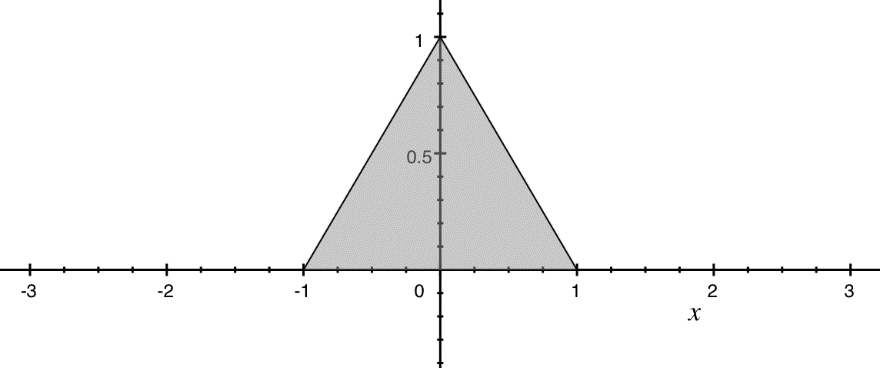
**Свертки (Convolution)**

Этот метод похож на аффинные преобразования тем, что используются фильтры, но имеет не фиксированное окно, а окно, пропорциональное масштабу. Например, если размер окна фильтра равен 6, а размер изображения уменьшается в 2,5 раза, то в формировании каждого пикселя конечного изображения принимает участие (2,5 \* 6)² = 225 пикселей. К счастью, свертки можно считать в 2 прохода, сначала в одну сторону, потом в другую, поэтому алгоритмическая сложность расчета каждого пикселя равна не 225, а всего (2,5 \* 6) \* 2 = 30. Вклад каждого исходного пикселя в конечный как раз определяется фильтром. Отношение обработанных пикселей к исходным целиком определяется размером окна фильтра и равно его квадрату. То есть для билинейного фильтра это отношение будет 4, для бикубического 16, для Ланцоша 36. **Алгоритм прекрасно работает как для уменьшения, так и для увеличения.**

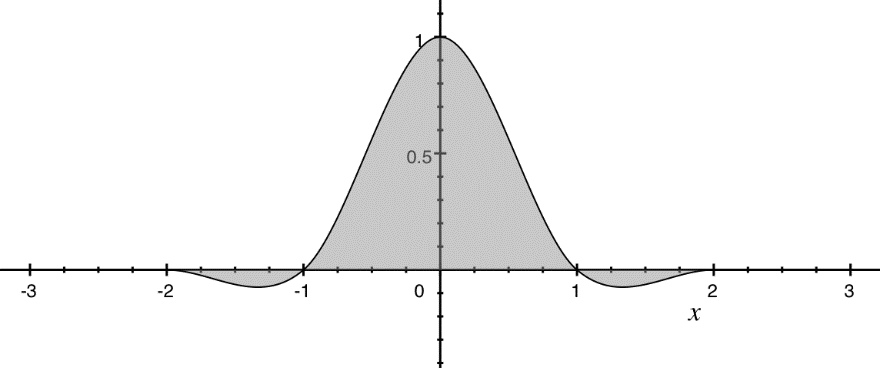
Скорость работы этого метода зависит от всех параметров: размеров исходного изображения, размера конечного изображения, размера окна фильтра.

В Матлабе поддерживаются следующие фильтры:

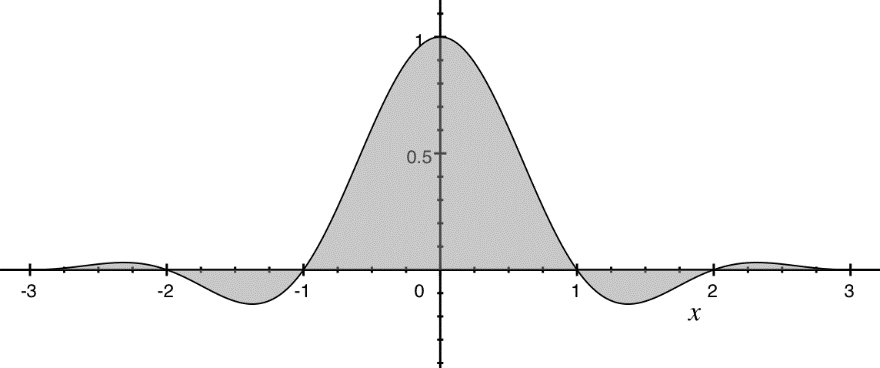
* Билинейный фильтр (параметр 'triangle'):



* Бикубический фильтр (параметр 'cubic'):



* Фильтр Ланцоша (параметры 'lanczos2' и 'lanczos3'):



Некоторые фильтры имеют зоны отрицательных коэффициентов (как например бикубический фильтр или фильтр Ланцоша). Это нужно для придания переходам на конечном изображении резкости, которая была на исходном.