آرش آژند 40003773

در این قسمت برای پیدا کردن مختصات پیکسل جدید در تصویر input image از **0.5** استفاده شده.

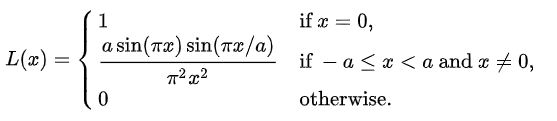
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| نام تصویر | **مقدار PSNR به ازای Resizing\_Factor برابر با 2** | | | | | | | |
| روش  bilinear | روش  Nearest  neighbor | روش استفاده از فاصله  Eucledian | روش استفاده از فاصله  City\_block | روش استفاده از فاصله  chessboard | روش bicubic | روش پیشنهادی  من | زمان اجرا بر حسب ثانیه |
| Boat | 29.1735 | 25.4745 | 29.1735 | 29.1736 | 29.1736 | 26.9338 | 29.2763 | 7.5543 |
| Peppers | 32.1940 | 28.1799 | 32.1940 | 32.1940 | 32.1940 | 29.7267 | 32.2311 | 2.0997 |
| Cameraman | 36.2901 | 28.0225 | 36.2900 | 36.2900 | 36.2900 | 30.4992 | 38.8513 | 2.0496 |
| House | 32.0683 | 27.5447 | 32.0678 | 32.0678 | 32.0676 | 29.2794 | 32.2597 | 0.5126 |
| متوسط PSNR | 32.4314 | 27.3054 | 32.4313 | 32.4313 | 32.4313 | 29.1 | 33.1546 | 3.0540 |

من در روش پیشنهادی از الگوریتم Lanczos interpolation استفاده کردم. این روش از یک کرنل اینترپولیشن به نام کرنل لانزوس استفاده می‌کند که بر اساس تابع سینک (sinc) و تابع پنجره‌ای لانزوس ساخته شده است.

این کرنل وزن‌ها را بر اساس فاصله از پیکسل مرکزی و با استفاده از تابع سینک تعدیل می‌کند، که به حفظ جزئیات و کاهش اثرات نامطلوب مانند تاری کمک می‌کند.

روش لانزوس معمولاً با استفاده از چندین لوب (معمولاً 2 یا 3 لوب) از تابع سینک انجام می‌شود، که هر لوب نمایانگر یک دوره از تابع سینک است، که در اینجا من از 2 لوب استفاده کردم.

فرمول کرنل لانزوس به اینصورت است:



پارامتر a همان پارامتر کرنل هست که در اینجا برابر با 2 هست و مشخص میکند چه تعداد همسایه در محاسبات در نظر گرفته شود.

با استفاده از این فرمول هم برای محاسبه ی مقدار پیکسل جدید در مکان دلخواه استفاده میشود.

این تابع را خود متلب هم در تابع imresize دارد ولی کاری ک من روی این تابع انجام دادم این بود که وقتی موقعیت پیکسل جدید را بدست می آورم با 0.5 جمع میکنم. انجام اینکار به این دلیل است که با اضافه کردن 0.5 به موقعیت پیکسل اجازه می دهد تا به نقطه مرکزی بین پیکسل های اصلی دست پیدا کند و چون میخواهیم با استفاده از همسایه های پیکسل جدید مقدار آن را مشخص کنیم، انجام اینکار باعث افزایش psnr و دقت تصویر جدید میشود.

برای مثال در تصویر “Pepper.png” و “LR\_Pepper.png” زمانی که تصویر کوچک شده را با استفاده از تابع متلب بزرگ میکنیم مقدار psnr با تصویر اصلی 29.7065 میشود در حالی که با استفاده از تابع با تغییرات خودم مقدار psnr با تصویر اصلی 32.2311 میشود.

* ایده ی دومم هم که بعد از این کد با کامنت مشخص کردم به اینصورت هست که ابتدا هر 4 همسایه پیکسل جدید بدست می آیند و میانگین سطح روشنایی این همسایه ها برای سطح روشنایی پیکسل جدید قرار داده میشود.

|  |  |  |
| --- | --- | --- |
| زمان اجرا بر حسب ثانیه | روش پیشنهادی دوم | نام تصویر |
| 1.1880 | 29.1735 | Boat |
| 0.0191 | 32.1940 | Peppers |
| 0.0169 | 36.2901 | Cameraman |
| 0.0075 | 32.0683 | House |
| 0.3078 | 32.4314 | متوسط PSNR |