Vol. 1 No. 4 (2022)
ISSN: 2181-3132
Yosh Tadqiqotchi Jurnali

OPENCL YORDAMIDA TASVIRLARGA GRAFIK ISHLOV BERISH BIRLIGI (GPU) DA QAYTA ISHLASH

Karimjonov Akmaljon Baxtiyor o'g'li

Muhammad Al-Xorazmiy nomidagi Toshkent axborot texnalogiyalari universtiteti, Magistranti

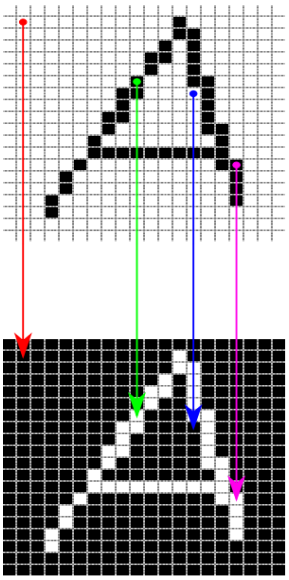
akmaljonkarimjanov@gmail.com

https://doi.org/10.5281/zenodo.6560845

Anotatsiya: Maqola OpenCl platformasida tasvirlarga qanday ishlov berish usullari va metodlarini o`z ichiga olgan.

Kalit so'zlar: Tasvir, piksel, Grafik ishlov berish birligi, yadro, trkstura.

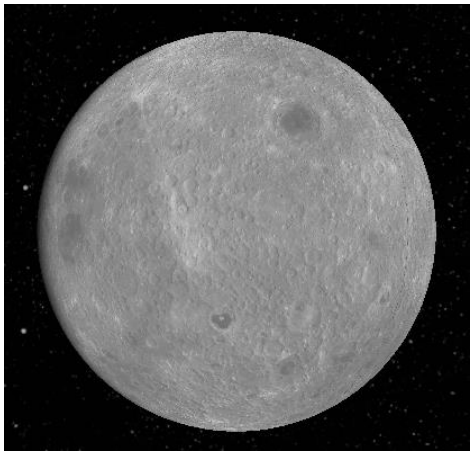
Tasvirni qayta ishlashning ko'p operatsiyalari tasvirdagi pikseldan pikselgacha takrorlanadi, joriy piksel qiymatidan foydalanib hisob-kitoblarni amalga oshiradi va nihoyat har bir hisoblangan qiymatni chiqish tasviriga yozadi. 1-rasmda misol sifatida kulrang qiymatni o'zgartirish operatsiyasi ko'rsatilgan.



1-rasm: Tasvir rangini o'zgartirish. Ba'zi bir joydagi chiqish pikselini faqat bir xil joydagi kirish pikselini hisobga olgan holda hisoblash mumkin. Ba'zi mos piksellar rangli chiziqlar bilan ko'rsatilgan.

Biz har bir pikselning kulrang qiymatini alohida o'zgartiramiz. Bu, shubhasiz, har bir piksel uchun bir vaqtning o'zida (parallel ravishda) amalga oshirilishi mumkin, chunki chiqish qiymatlari bir-biriga bog'liq emas.

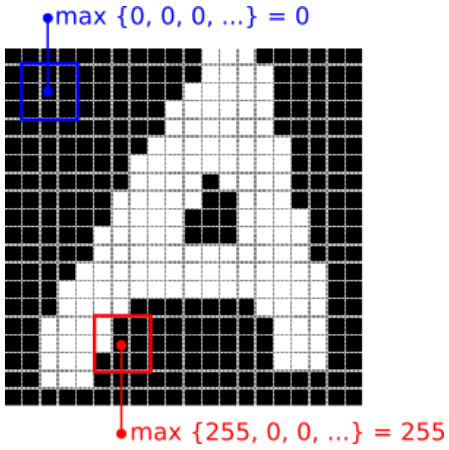
Rasmga ishlov berishda biz piksel qiymatlariga tezkor kirishimiz kerak. GPU'lar grafik maqsadlar uchun mo'ljallangan va ulardan biri teksturalashdir, shuning uchun piksellarga kirish va boshqarish uchun apparat yaxshi optimallashtirilgan. 2-rasmda 3D-dasturlarda tekstura odatda qanday maqsadda qo'llanilishi ko'rsatilgan.



2-rasm: Teksturalar ko'pincha modellarni (bu holda shar shaklida) yanada realroq qilish uchun ishlatiladi.

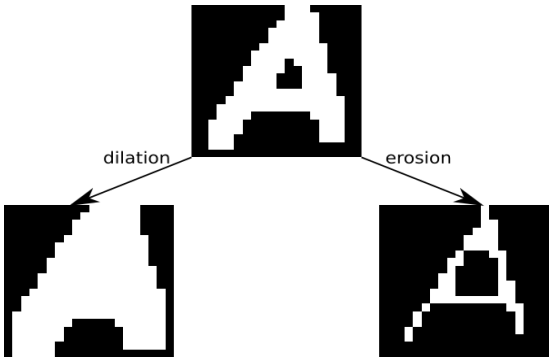
Kengayish va eroziya.

Keling, biz amalga oshiradigan tasvirni qayta ishlash usullarini qisqacha ko'rib chiqaylik. Har ikkala amal ham morfologik amallar sinfiga kiradi. Niqob (ko'pincha tuzilish elementi deb ataladi) kirish tasviri bo'ylab piksel piksel siljiydi. Har bir joy uchun niqob ostidagi maksimal (kengayish uchun) yoki minimal (eroziya uchun) piksel qiymati chiqish tasviriga yoziladi. 3-rasmda kengayish qanday hisoblanganligi tasvirlangan.



3-rasm: 3×3 niqob tasvir bo‘ylab siljiydi. Kengayish uchun niqob ostidagi maksimal piksel qiymati natijaviy qiymat sifatida olinadi. Niqobning ikkita joyi ko'rsatilgan, biri qora pikselga (0), ikkinchisi oq pikselga (255) olib keladi.

Ikkala operatsiya haqida tasavvurga ega bo'lish uchun 4-rasmga qarang. Bundan tashqari, 5-rasmda kulrang qiymatli tasvir va uning eroziyalangan versiyasi ko'rsatilgan.



4-rasm: Yuqori: kiritilgan rasm. Pastki: kengayish va eroziya uchun tasvirlarni chiqarish.

5-rasm: Ushbu misol bir nechta eroziya yordamida qalam kengligini qanday oshirishni ko'rsatadi.

GPUlarning parallelligidan foydalanish.

Ko'rib turganingizdek, kengayish yoki eroziyani qo'llashda niqob tasvir bo'ylab siljiydi. Har bir joyda bir xil asosiy operatsiya qo'llaniladi (niqob bo'yicha piksellar to'plamini tanlash, maksimal yoki minimal piksel qiymatini chiqarish). Bundan tashqari, chiqish piksellari bir-biriga bog'liq emas. Bu joylashuv uchun qo'shimcha argument bilan kirish tasviridagi funktsiya sifatida asosiy operatsiyani amalga oshirishni taklif qiladi. Keyin, chiqish tasvirini

olish uchun bir vaqtning o'zida (parallel ravishda) barcha mumkin bo'lgan joylar uchun funktsiyani chaqiramiz (x 0 va kenglik-1 orasida, y 0 va balandlik-1 orasida).

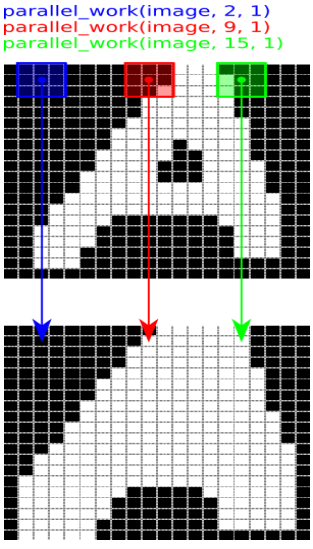
Avval psevdo-kodni amalga oshirishni amalga oshiramiz: yoki biz sinxron_ish (tasvir) deb nomlaymiz va har bir iteratsiya bosqichida bir xil amalni qo'llagan holda barcha (x, y) joylarda takrorlaymiz. Yoki parallel_work(tasvir, x, y) ni barcha parallel joylashuvlar uchun qo'shimcha joylashuv argumentlari bilan chaqiramiz (masalan, har bir misol uchun ipni boshlash orqali). 6-rasmda chiqish piksellari qiymatlari parallel ravishda qanday hisoblanganligi ko'rsatilgan.

Sinxron amalga oshirish:

synchronous_work(image):
for all (x, y) locations in image:
1. select pixels from 3×3 neighborhood of (x, y)
2. take maximum (or minimum)
3. write this value to output image at (x, y)in main: run one instance of synchronous_work(image)

Parallel amalga oshirish:

parallel_work(image, x, y):
1. select pixels from 3×3 neighborhood of (x, y)
2. take maximum (or minimum)
3. write this value to output image at (x, y)in main: run one instance of parallel_work(image, x, y) for each (x, y) location in the image at the same time



6-rasm: Niqobning uchta joyi ko'rsatilgan. Olingan piksel qiymatlari bir-biriga bog'liq emas va shuning uchun bir vaqtning o'zida hisoblanishi mumkin.

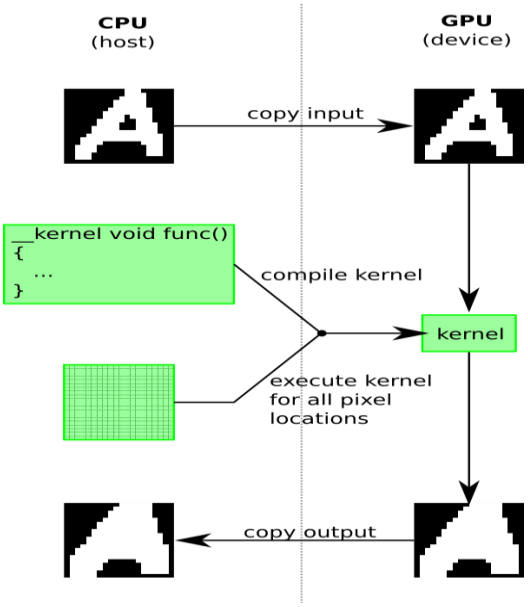
OpenCL amalga oshirish.

Keling, psevdokodimizni haqiqiy grafik protsessorda ishlashi uchun portga kiritamiz: biz Open Computing Language (OpenCL) ramkasidan foydalanamiz. 7-rasmda bizning amalga oshirishimiz nima qilishini ko'rsatadi: biz birinchi navbatda kiritilgan rasmimizni GPUga ko'chiramiz, yadroni (GPU dasturini) kompilyatsiya qilamiz, uni barcha piksel joylari uchun parallel ravishda bajaramiz va nihoyat olingan tasvirni GPUdan qayta nusxalaymiz.

Xost uchun bitta dastur va qurilma uchun bitta dasturni amalga oshirishimiz kerak:

CPU ("host"): OpenCL-ni o'rnatadigan, CPU va GPU o'rtasida ma'lumotlarni uzatishni amalga oshiradigan, fayllarni qayta ishlash bilan shug'ullanadigan Python dasturi va hokazo.

GPU ("qurilma"): C-ga o'xshash til yordamida tasvirni haqiqiy qayta ishlashni amalga oshiradigan OpenCL yadrosi.



Foydalanilgan adabiyotlar:

The OpenCL Specification, 2012 Khronos OpenCL Working Group.

[1] Scarpino — OpenCL in Action

Websayt: https://www.khronos.org/opencl/