OPTIMAL MULTIDIMENSIONAL CYCLIC CONVOLUTION ALGORITHMS FOR DEEP LEARNING AND COMPUTER VISION APPLICATIONS

Prof. Ioannis Pitas

pitas@csd.auth.gr
Department of Informatics, Aristotle University of Thessaloniki
Thessaloniki 54124, Greece

ABSTRACT

1D, 2D and multidimensional convolutions are basic tools in deep learning, notably in convolutional neural networks (CNNs) and in computer vision (template matching, correlation trackers). Therefore, fast 1D/2D/3D convolution algorithms are essential for advanced machine learning and computer vision. This paper presents: 1) novel optimal n-D cyclic convolution algorithms having minimal multiplicative complexity that are much faster than any competing convolution algorithm internationally and 2) methods for speeding up such optimal convolution algorithms on GPUs and multicore CPUs. Such a speedup is very important both for CNN training and CNN testing, particularly in embedded environments (e.g., on drones) and real-time applications (e.g., fast CNN inference for object detection and correlation trackers for embedded real-time object tracking).

Index Terms— Convolutional Neural Networks, Fast convolutions

1. INTRODUCTION

2D convolutional layers in CNNs [1] typically convolve the feature map $\mathbf{X}_l$ (3D tensor of dimensions $N_l \times M_l \times C_l$) of the neural network layer l with a k-th convolution kernel $\mathbf{w}_{l,k}$ (3D tensor of dimensions $H_{l,k} \times W_{l,k} \times C_l$), add a bias term $b(l,k)$ and then pass it through a nonlinearity activation function f, (e.g., RELU), to produce the feature map $\mathbf{X}_{l+1,k}$ (3D tensor of dimensions $N_{l+1,k} \times M_{l+1,k} \times C_{l+1}$) of layer $l+1$:

$$x(i,j,c_{l+1},l+1,k) = f(b(l,k)+ \sum_{c=1}^{C_l} \sum_{i'=0}^{H_{l,k}} \sum_{j'=0}^{W_{l,k}} h(i',j',l,k)x(i-i',j-j',c,l,k)). \quad (1)$$

The extension to higher spatial or spatiotemporal dimensions is straightforward. It is essential to devise fast 2D and multidimensional convolution algorithms, in order to have fast CNN training and testing. Furthermore, the same algorithms can be used for calculating the 2D correlation of an input image x) and a template h, e.g., for fast correlation trackers [2], [3].

The construction of fast convolution algorithms is a heavily researched topic in the signal processing community. It reached maturity in the 90ties [4], [5], [6].

Recently, a resurgence of fast linear convolution algorithms for CNNs occured, collectively called Winograd convolutions [7], [8]. Various implementations for GPUs and multicore CPUs appeared [7], [9] and numerical stability issues have been investigated. However, most of the recent algorithms are suboptimal.

2. FAST 2D AND MULTIDIMENSIONAL CONVOLUTION ALGORITHMS WITH MINIMAL COMPUTATIONAL COMPLEXITY

A linear convolution of signal x having length L with a convolutional kernel h having length M produces an output signal $y(n) = x(n) * h(n)$ of length $L+M-1$ and can be embedded in a cyclic convolution of length $N \geq L+M-1$, by zero padding both the signal x and convolution kernel h. The following relation holds for the Z-transform of an 1D cyclic convolution of length N:

$$y(n) = x(n) \circledast h(n) <=> Y(z) = X(z)H(z) \mod (Z^N - 1). \quad (2)$$

The 1D cyclic Winograd convolution algorithms are proven to be of the form:

$$\mathbf{y} = \mathbf{C}(\mathbf{Ax} \otimes \mathbf{Bh}). \quad (3)$$

This fast convolution architecture is shown in Figure 1.

In this section, we shall focus on fast 2D cyclic convolution:

$$y(k_1,k_2) = \sum_{i_1}^{N_1} \sum_{i_2}^{N_2} h(i_1,i_2)x((k_1-i_1)_{N_1},(k_2-i_2)_{N_2})) \quad (4)$$

algorithms having minimal computational complexity, as the methodology for fast multidimensional convolutions is similar [5], [6]. These convolutions have the following form in

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 871479 (AERIAL-CORE).

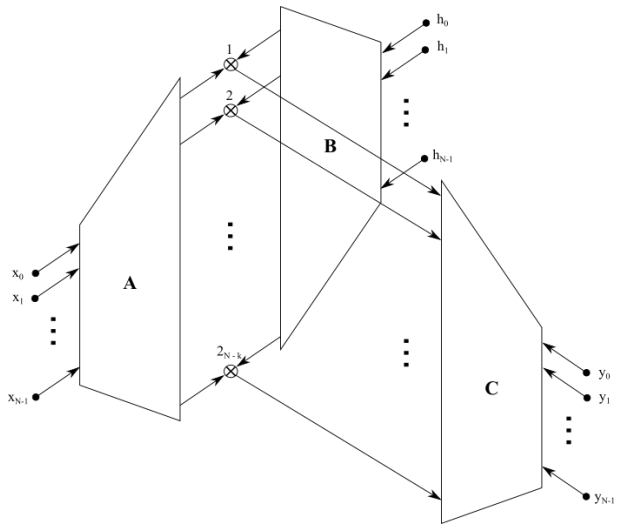

Fig. 1: 1D Winograd cyclic convolution algorithm.

the 4D Z domain (z_1, z_2):

$$Y(z_1, z_2) = X(z_1, z_2)H(z_1, z_2) \mod ({z_1}^{N_1} - 1), ({z_2}^{N_2} - 1), \tag{5}$$

or more generally:

$$Y(z_1, z_2) = X(z_1, z_2)H(z_1, z_2) \mod P_1(z_1), P_2(z_2) \tag{6}$$

If $P_i(z_i), i = 1, 2$ factorize as follows:

$$P_i(z_i) = \prod_{j_i=1}^{\nu_i} P_{ij_i}(z_i), \quad 1 \le j_i \le \nu_i, \quad i = 1, 2, \tag{7}$$

the convolution (6) is split into $\nu_1\nu_2$ smaller products:

$$Y_{1j_1,2j_2}(z_1, z_2) = X_{1j_1,2j_2}(z_1, z_2)H_{1j_1,2j_2}(z_1, z_2) \tag{8}$$

$$\mod P_{1j_1}(z_1), P_{2j_2}(z_2), 1 \le j_i \le \nu_i, i = 1, 2. \tag{9}$$

By finding the polynomials:

$$R_{ij_i}(z_i) = \delta_{j_i,k_i} \mod P_{ij_i}(z_i), P_{ik_i}(z_i) \quad 1 \le j_i \le \nu_i, k_i \le \nu_i, \quad i = 1, 2, \tag{10}$$

we can reconstruct $Y(z_1, z_2)$ as follows [6]:

$$Y(z_1, z_2) = \sum_{j_1=1}^{\nu_1} Y_{1j_1,2j_2}(z_1, z_2) \sum_{j_2=1}^{\nu_2} R_{1j_1}(z_1)R_{2j_2}(z_2) \mod P_1(z_1), P_2(z_2). \tag{11}$$

The algorithm (6)-(11) is essentially the split nesting convolution algorithm [4], which is a variation of the nesting algorithm of [10].

The computational complexity of this 2D convolution algorithm is $(2N_1-\nu_1)(2N_2-\nu_2)$, which is $O(N^2)$, i.e., much lower than the computational complexity of $O(N^4)$ of the 2D cyclic convolution computation by its definition (4). However, this is not the minimal computational complexity algorithm, as: a) each polynomial $P_{1j_1}(z_1), 1 \le j_1 \le \nu_1$ can possibly be further factorized in k_{j_1,j_2} factors over the field $Q[z_2]/P_{2j_2}(z_2), 1 \le j_2 \le \nu_2$ or b) vice versa, each polynomials $P_{2j_2}(z_2), 1 \le j_2 \le \nu_2$ can possibly be further factorized k_{j_2,j_1} factors over the fields $Q[z_1]/P_{1j_1}(z_1), 1 \le j_1 \le \nu_1$. By examining both these further factorizations (a, b) for each product (8), we can derive 2D cyclic convolution algorithms having minimal computational complexity [6]:

$$M = \sum_{j_1=1}^{\nu_1} \sum_{j_2=1}^{\nu_2} min((2N_{j_2} - 1)(2N_{j_1} - k_{j_1,j_2}), (2N_{j_1} - 1)(2N_{j_2} - k_{j_2,j_1})) \tag{12}$$

Such algorithms take the general form of (3). However, the construction of matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ requires good Algebra skills and is far from trivial. We constructed such novel optimal algorithms for several $N \times N$ cases, notably for $p \times p$ (e.g., for $p = 3, 5, 7$) and for $2^l \times 2^l$ (e.g., for 4×4) cyclic convolutions. An illustrative example of this procedure for a 3×3 cyclic convolution can be found in the next section.

CNNs typically employ 2D linear convolutions having small convolution kernels (e.g., below 11×11 coefficients). They, in turn, can be embedded in rather small 2D $N \times N$ cyclic convolution, as input images are typically split into small blocks to enable block-based 2D $N \times N$ cyclic convolution calculations that are highly (and easily) parallelizable in GPUs, as can be seen in a subsequent section.

3. EXAMPLE: FAST 2D 3×3 CYCLIC CONVOLUTION ALGORITHM HAVING MINIMAL COMPUTATIONAL COMPLEXITY

A 2D 3×3 cyclic convolution is defined as follows:

$$Y(z_1, z_2) = H(z_1, z_2)X(z_1, z_2) \mod z_1^3 - 1, z_2^3 - 1, \tag{13}$$

where:

$$X(z_1, z_2) = x_{00} + x_{01}z_2 + x_{02}z_2^2 + x_{10}z_1 + x_{11}z_1z_2 + x_{12}z_1z_2^2 + x_{20}z_1^2 + x_{21}z_1^2z_2 + x_{22}z_1^2z_2^2 \tag{14}$$

$$H(z_1, z_2) = h_{00} + h_{01}z_2 + h_{02}z_2^2 + h_{10}z_1 + h_{11}z_1z_2 + h_{12}z_1z_2^2 + h_{20}z_1^2 + h_{21}z_1^2z_2 + h_{22}z_1^2z_2^2. \tag{15}$$

The polynomial $z^3 - 1$ is analyzed as follows:

$$z^2 - 1 = (z - 1)(z^2 + z + 1). \tag{16}$$

Therefore, the 2D 3×3 cyclic convolution is decomposed as follows:

$$X_1(z_1, z_2) = X(z_1, z_2) \mod (z_1 - 1), (z_2 - 1) \quad (17)$$

$$X_2(z_1, z_2) = X(z_1, z_2) \mod (z_1 - 1)(z_2^2 + z_2 + 1) \quad (18)$$

$$X_3(z_1, z_2) = X(z_1, z_2) \mod (z_2 - 1)(z_1^2 + z_1 + 1) \quad (19)$$

$$X_4(z_1, z_2) = X(z_1, z_2) \mod (z_1^2 + z_1 + 1)(z_2^2 + z_2 + 1). \quad (20)$$

We notice that the polynomial $z_1^2 + z_1 + 1$ can be factorized in the field $Q[z_2]/z_2^2 + z_2 + 1$ as follows:

$$z_1^2 + z_1 + 1 = (z_1 - z_2)(z_1 + 1 + z_2). \quad (21)$$

Thus, $Y_4(z_1, z_2)$ can be further decomposed in two terms:

$$X_{4_1}(z_1, z_2) = X_4(z_1, z_2) \mod (z_1 - z_2)(z_2^2 + z_2 + 1) \quad (22)$$

$$X_{4_2}(z_1, z_2) = X_4(z_1, z_2) \mod (z_1 + z_2 + 1)(z_2^2 + z_2 + 1) \quad (23)$$

By employing CRT, $Y_4(z_1, z_2)$ is reconstructed from $Y_{4_1}(z_1, z_2)$ and $Y_{4_2}(z_1, z_2)$ as follows:

$$\begin{aligned} Y_4(z_1, z_2) &= \sum_{n=1}^{2} R_i(z_1, z_2) Y_{4_i}(z_1, z_2) \\ \mod & \quad (z_1^2 + z_1 + 1)(z_2^2 + z_2 + 1) \end{aligned} \quad (24)$$

$$R_1(z_1, z_2) = -\frac{1}{3}[2z_1 z_2 + z_2 + 1] \quad (25)$$

$$R_2(z_1, z_2) = \frac{1}{3}[(2z_2 + 1)z_1 + z_2 + 2]. \quad (26)$$

Then $Y(z_1, z_2)$ is reconstructed as follows:

$$Y(z_1, z_2) = \sum_{i=1}^{4} R_i(z_1, z_2) Y_i(z_1, z_2) \mod (z_1^3 - 1)(z_2^3 - 1), \quad (27)$$

where:

$$R_1(z_1, z_2) = \frac{1}{9}(z_1^2 + z_1 + 1)(z_2^2 + z_2 + 1) \quad (28)$$

$$R_2(z_1, z_2) = -\frac{1}{9}(z_1^2 + z_1 + 1)(z_2^2 + z_2 - 2) \quad (29)$$

$$R_3(z_1, z_2) = -\frac{1}{9}(z_1^2 + z_1 - 2)(z_2^2 + z_2 + 1) \quad (30)$$

$$R_4(z_1, z_2) = \frac{1}{9}(z_1^2 + z_1 - 2)(z_2^2 + z_2 - 2). \quad (31)$$

This leads to fast 2D 3×3 cyclic convolution algorithm of the form:

$$\mathbf{y} = \mathbf{C}(\mathbf{A}\mathbf{x} \otimes \mathbf{B}\mathbf{h}), \quad (32)$$

where:

$$\mathbf{A} = \mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & -1 & 1 & 0 & -1 & 1 & 0 & -1 \\ 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 \\ 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & -1 & -1 & -1 \\ 0 & 0 & 0 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & 1 & -1 & -1 & -1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & -1 & 1 & -1 & 1 & 0 \\ 0 & 1 & -1 & 1 & -1 & 0 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 & 0 & 1 & 0 & 1 & -1 \\ 1 & 0 & -1 & -1 & 1 & 0 & 0 & -1 & 1 \\ 0 & 1 & -1 & -1 & 0 & 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 0 & 1 & -1 & -1 & 0 & 1 \end{bmatrix}, \quad (33)$$

This fast algorithm requires only 13 multiplications, while the computation of the 3×3 cyclic convolution using its definition (13) requires 81 multiplications. Also note that the two matrix-vector products and the point-wise vector product are highly parallelizable. Furthermore, as expected, matrix $\mathbf{A}$, $\mathbf{B}$ entries are $0, \pm 1$. Therefore, if we use the form (3), we have no 'multiplications in the matrix-vector products. Furthermore, additions/subtractions used in the matrix vector products, e.g., $\mathbf{X} = \mathbf{A}\mathbf{x}$ can be grouped in subsums that can reused. In the case of the $\mathbf{A}\mathbf{x}$ computation, the additions can be reduced from 68 to 40, as can be seen in the flow diagram of Figure 2. The constructions of algorithms of the form (3), taking all these optimizations into account, is novel, does pay off and it is far from trivial.

Transformation of each one of the 13 rows of matrix $\mathbf{A}$ into 3×3 submatrices leads to interesting visualization patterns, as can be seen in Figure 3. Essentially, each matrix row (but for the first one) produces an input 2D signal (image) $\mathbf{x}$ transformation on certain directions and frequency bands.

4. SPEEDING UP AND PARALLELIZATION OF CONVOLUTION ALGORITHMS

In case $L >> M$, the signal x can be split in blocks of length L_B. Then the linear convolution can be split in smaller length $N \geq L_B + M - 1$ convolutions using overlap-add or overlap-save methods [11]. This approach leads to very easily parallelizable convolution algorithms.

The proposed 2D convolution method was implemented on GPU cards for a 3×3 convolution kernel on 512×512 pixel input images, using an optimal 4×4 cyclic convolution algorithm combined with an overlap-save block-based approach employing 65536 2×2 image blocks (tiles). Its execution time was a mere 0.0809 ms. It is 4,77 times faster than the fastest cuDNN convolution (GEMM-0) and 11,33 times faster than the corresponding cuDNN Winograd linear convolution rou-

$$\mathbf{C} = \frac{1}{27}\begin{bmatrix} 3 & 3 & -6 & 3 & 3 & -6 & 3 & 3 & -6 & 3 & 3 & -6 & 3 \\ 3 & 3 & 3 & -6 & 3 & -6 & 3 & 3 & 3 & -6 & 3 & 3 & -6 \\ 3 & -6 & 3 & 3 & 3 & -6 & 3 & -6 & 3 & 3 & -6 & 3 & 3 \\ 3 & 3 & -6 & 3 & 3 & 3 & -6 & 3 & 3 & -6 & -6 & 3 & 3 \\ 3 & 3 & 3 & -6 & 3 & 3 & -6 & -6 & 3 & 3 & 3 & -6 & 3 \\ 3 & -6 & 3 & 3 & 3 & 3 & -6 & 3 & -6 & 3 & 3 & 3 & -6 \\ 3 & 3 & -6 & 3 & -6 & 3 & 3 & -6 & 3 & 3 & 3 & 3 & -6 \\ 3 & 3 & 3 & -6 & -6 & 3 & 3 & 3 & -6 & 3 & -6 & 3 & 3 \\ 3 & -6 & 3 & 3 & -6 & 3 & 3 & 3 & 3 & -6 & 3 & -6 & 3 \end{bmatrix}. \quad (34)$$

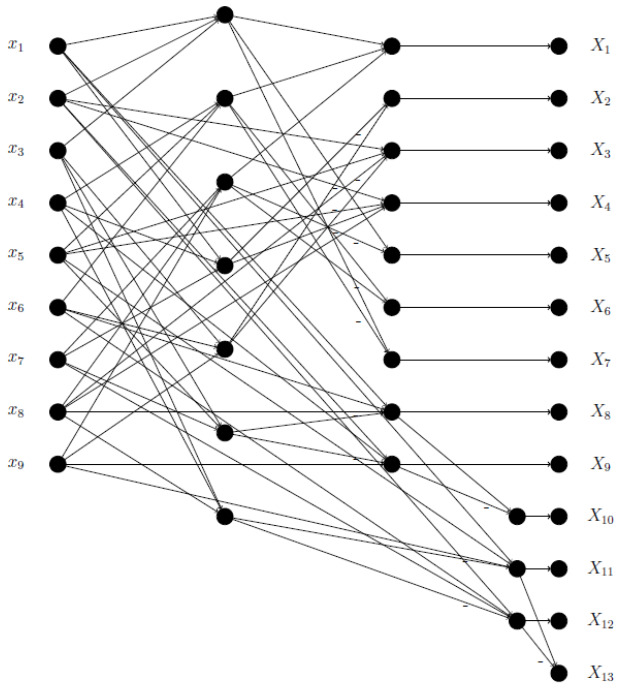

Fig. 2: Diagram for $\mathbf{Ax}$ calculation for a 3×3 Winograd cyclic convolution.

tine (Winograd-6) for the same convolution kernel size and image size. Therefore, the proposed algorithm is almost 5 times faster than the fastest known competing convolution algorithm internationally.

5. CONCLUSIONS

CNN are still slow to be used, e.g., for object detection in embedded vision systems [12]. Furthermore, R & D on CNNs moves towards higher dimensions for 3D spatial or spatiotemporal (video) analysis, where processing requirements are excessive. Finally, fast convolution structures and essential for fast object tracking (e.g., correlation trackers) in embedded systems [3]. Therefore, it indeed pays off to derive fast (possibly optimal) multidimensional convolution algorithms.

This paper presents novel 2D and n-D cyclic convolution algorithms having minimal computational complexity. They can be easily described by simple linear algebra operations with matrices having trivial entries $0, \pm 1$. Their structure is easily paralellizable. This renders these algorithms very attractive for multicore CPU and GPU processing. They are much faster than any competing convolution algorithm known today. However, they do have their drawbacks. In the general case of 2D $N \times N$ convolution, their structure cannot easily be obtained and requires strong mathematical skills. The process of automatically generating the matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ to be used in (3) is an active and very interesting research topic.

6. REFERENCES

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436, 2015.

[2] Joao F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[3] Paraskevi Nousi, Danai Triantafyllidou, Anastasios Tefas, and Ioannis Pitas, "Joint lightweight object tracking and detection for unmanned vehicles," in *Proceed-*

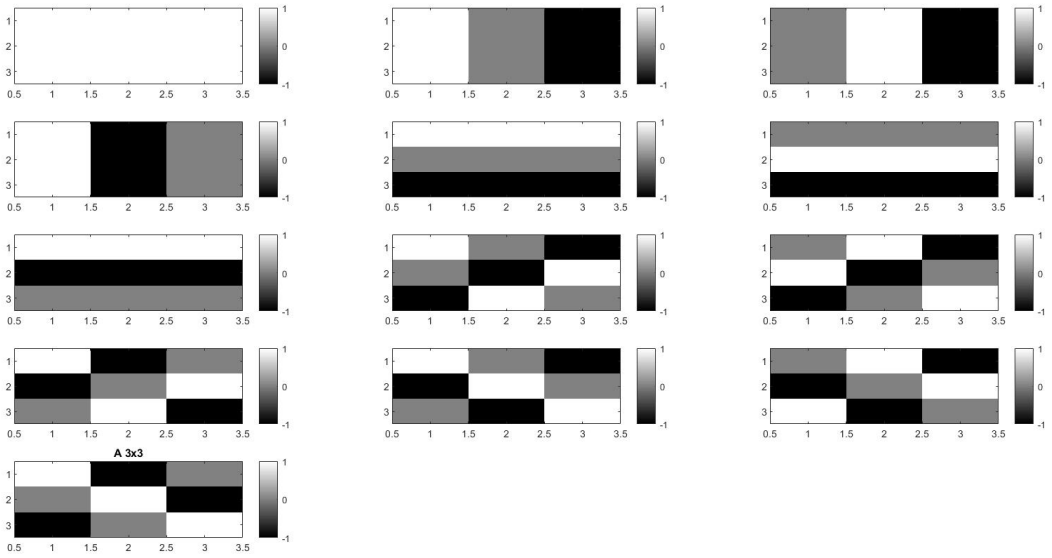

Fig. 3: Visualization of rows of the 3×3 Winograd cyclic convolution matrix $\mathbf{A}$.

ings of the of the IEEE International Conference on Image Processing (ICIP), 2019.

[4] Henri J Nussbaumer, *Fast Fourier transform and convolution algorithms*, Springer Science & Business Media, 1981.

[5] I Pitas, *Computational complexity study of multidimensional digital signal processing algorithms*, Aristotle University of Thessaloniki, Greece, 1985.

[6] Ioannis Pitas and M Strintzis, "Multidimensional cyclic convolution algorithms with minimal multiplicative complexity," *IEEE transactions on acoustics, speech, and signal processing*, vol. 35, no. 3, pp. 384–390, 1987.

[7] Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer, "cudnn: Efficient primitives for deep learning," *arXiv preprint arXiv:1410.0759*, 2014.

[8] Andrew Lavin and Scott Gray, "Fast algorithms for convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4013–4021.

[9] Zhen Jia, Aleksandar Zlateski, Fredo Durand, and Kai Li, "Optimizing n-dimensional, winograd-based convolution for manycore cpus," in *Proceedings of the 23rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*. ACM, 2018, pp. 109–123.

[10] R Agarwal and J Cooley, "New algorithms for digital convolution," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 5, pp. 392–410, 1977.

[11] Ioannis Pitas, *Digital image processing algorithms and applications*, John Wiley & Sons, 2000.

[12] N.Nikolaidis I. Mademlis, V. Mygdalis and I.Pitas, "Challenges in autonomous uav cinematography: An overview," *IEEE International Conference on Multimedia and Expo (ICME)*, pp. 392–410, 2018.