Algorithm 1 ptmKernel3(IN/OUT: r; IN: $c0, c2, c4, c6, omega$)

1: $threadX \leftarrow blockDim.x \cdot blockIdx.x + threadIdx.x$;
2: $threadZ \leftarrow blockDim.z \cdot blockIdx.z + threadIdx.z$;
3: $idx\_x \leftarrow threadX + 1$;
4: $idx\_z \leftarrow threadZ + 1$;
5: $currentY \leftarrow 1$;
6: **for** $s \in [3; GridNx + GridNy + GridNz - 3]$ **do**
7: **if** $(idx\_x + currentY + idx\_z = s) \wedge (s < GridNy + idx\_x + idx\_z)$ **then**
8: $nodeIndex \leftarrow idx\_x + (BlockDimX + 1) \cdot currentY + GridXY \cdot idx\_z$;
9: $m0 \leftarrow nodeIndex$;
10: $c0m0 \leftarrow c0[m0]$;
11: **if** $c0m0 > 0$ **then**
12: $m2 \leftarrow m0 - 1$;
13: $m4 \leftarrow m0 - GridNx$;
14: $m6 \leftarrow m0 - GridXY$;
15: $rm4 \leftarrow 0$;
16: **if** $(s > 3 + threadX + threadZ)$ **then**
17: $rm4 \leftarrow cache[threadX][threadZ]$;
18: **else**
19: $rm4 \leftarrow r[m4]$;
20: $rm2 \leftarrow 0$;
21: **if** $(threadX \neq 0) \wedge (s > 3 + threadX + threadZ)$ **then**
22: $rm2 \leftarrow cache[threadX - 1][threadZ]$;
23: **else**
24: $rm2 \leftarrow r[m2]$;
25: $rm6 \leftarrow 0$;
26: **if** $(threadZ \neq 0) \wedge (s > 3 + threadX + threadZ)$ **then**
27: $rm6 \leftarrow cache[threadX][threadZ - 1]$;
28: **else**
29: $rm6 \leftarrow r[m6]$;
30: $rm0 \leftarrow (omega \cdot (c2[m0] \cdot rm2 + c4[m0] \cdot rm4 + c6[m0] \cdot rm6) + r[m0])/((0.5 \cdot omega + 1) \cdot c0m0)$;
31: $cache[threadX][threadZ] \leftarrow rm0$;
32: $r[m0] \leftarrow rm0$;
33: $currentY \leftarrow currentY + 1$;