In[3]:=
```
SetOptions[SelectedNotebook[],
 PrintingStyleEnvironment → "Printout", ShowSyntaxStyles → True]

(* GridTraps_RandomPhase *)

(* Firstly *)
p = na * nb;  (* the number of traps *)
na = 5; (*number of gridtraps in along x *)
nb = 5; (* number of gridtraps in along y*)
Da = 4;(*width along x*)
Db = 4; (*width along y*)
n = 512;(* n*n Grid *)

(*Secondly, list of position of traps,
list of phase of traps, and Matrix of Traps *)
        (*list of position of traps *)

For[i = 1; GridTraps = {}, i ≤ na, i++,
  For[j = 1, j ≤ nb, j++, elem = {-((na + 1)/2) * Da + i * Da, -((nb + 1)/2) * Db + j * Db};
   GridTraps = Append[GridTraps, elem]]
 ];

        (* Matrix of Traps *)
TrapsWithoutPhase :=
  Table[Sum[If[(x - GridTraps[[index, 1]] == 0) && (y - GridTraps[[index, 2]] == 0), 1, 0],
    {index, 1, p}], {x, -n / 2 + 1, n / 2}, {y, -n / 2 + 1, n / 2}];


(* Thirdly, define useful command: intensity,
etc. and Fourier transform and its inverse *)
Intensity[f_] := Re[f * Conjugate[f]];

DPlot[f_] := ListDensityPlot[Intensity[f], PlotRange → All];

(* take inverse Fourier transform of traps*)
IVFT[h_] := Module[{calcPattern}, calcPattern = InverseFourier[h];
   quad1 = Take[calcPattern, {1, Round[n / 2]}, {Round[n / 2] + 1, n}];
quad2 = Take[calcPattern, {1, Round[n / 2]}, {1, Round[n / 2]}];
quad3 = Take[calcPattern, {Round[n / 2] + 1, n}, {1, Round[n / 2]}];
quad4 = Take[calcPattern, {Round[n / 2] + 1, n}, {Round[n / 2] + 1, n}];
lM = Join[Flatten[quad4], Flatten[quad1]];
```

```
lftMtrx = Partition[lM, Round[n / 2]];
rM = Join[Flatten[quad3], Flatten[quad2]];
rghtMtrx = Partition[rM, Round[n / 2]];
calcPattern = Partition[
      Join[Flatten[Transpose[lftMtrx]], Flatten[Transpose[rghtMtrx]]], n]];
(* After taking the inverse Fourier transform, we want only the phase. So,
we are gonna extract the phase from the actual matrix and have each spot on the
 matrix have the same amplitude of electric field(1) with the phase extracted *)
HolPhaseOnly[realhol_] := Sqrt[p] / n * Exp[I Arg[realhol]];
(* the constant in front of the exponential is such that the total
 intensity is still the same as the first traps pattern,namely, p*)

(* Then,
we are gonna take the Fourier Transform of the phase-extracted hologram(g) *)
WFT[g_] := Module[{calcPattern, returnPattern}, calcPattern = g;
   quad1 = Take[calcPattern, {1, Round[n / 2]}, {Round[n / 2] + 1, n}];
quad2 = Take[calcPattern, {1, Round[n / 2]}, {1, Round[n / 2]}];
quad3 = Take[calcPattern, {Round[n / 2] + 1, n}, {1, Round[n / 2]}];
quad4 = Take[calcPattern, {Round[n / 2] + 1, n}, {Round[n / 2] + 1, n}];
lM = Join[Flatten[quad4], Flatten[quad1]];
lftMtrx = Partition[lM, Round[n / 2]];
rM = Join[Flatten[quad3], Flatten[quad2]];
rghtMtrx = Partition[rM, Round[n / 2]];
calcPattern = Partition[
      Join[Flatten[Transpose[lftMtrx]], Flatten[Transpose[rghtMtrx]]], n];
   returnPattern = Fourier[calcPattern]];


(* Statistics command *)
(* TotalInt[matrix_]:=Sum[Intensity[(Flatten[matrix])[[i]]],{i,1,n^2}]; *)
(* note that total intensity will be just n^2 for the phase-extracted hologram *)


TrapsIntTable[matrix_, position_] := Table[
   Intensity[matrix[[position[[i, 1]] + n/2, position[[i, 2]] + n/2]]] +
    Intensity[matrix[[position[[i, 1]] + n/2 + 1, position[[i, 2]] + n/2 + 1]]] +
    Intensity[matrix[[position[[i, 1]] + n/2 + 1, position[[i, 2]] + n/2]]] +
    Intensity[matrix[[position[[i, 1]] + n/2 + 1, position[[i, 2]] + n/2 - 1]]] +
```

```
    Intensity[matrix[[position[[i, 1]] + n/2, position[[i, 2]] + n/2 + 1]]] +
    Intensity[matrix[[position[[i, 1]] + n/2, position[[i, 2]] + n/2 - 1]]] +
    Intensity[matrix[[position[[i, 1]] + n/2 - 1, position[[i, 2]] + n/2 + 1]]] +
    Intensity[matrix[[position[[i, 1]] + n/2 - 1, position[[i, 2]] + n/2]]] +
    Intensity[matrix[[position[[i, 1]] + n/2 - 1, position[[i, 2]] + n/2 - 1]]], {i, 1, p}];

SumInt[f_] := Sum[f[[i]], {i, 1, p}];(* not really use in here*)

DifEff[IntenList_] := SumInt[IntenList] / p;(*1→perfect efficiency, 0→ worst*)


sd[IntenList_] := StandardDeviation[IntenList];


Uniformity[IntenList_] := 1 - (Max[IntenList] - Min[IntenList])/(Max[IntenList] + Min[IntenList]);
(* maybe we want to see how uniform the magnitude of the ideal hologram is. So,
use this function*)
DeviationAll[RealholMatrix_] :=
  StandardDeviation[Intensity[Flatten[RealholMatrix]]];

(* end of commands *)


(*OutputCommand *)
a = GridTraps; Print[a];
c = TrapsWithoutPhase;
d = IVFT[c];
e = HolPhaseOnly[d];
f = WFT[e];
g = DeviationAll[d]; Print[g];
h = TrapsIntTable[f, a]; Print[h];
i = DifEff[h]; Print[i];
j = sd[h]; Print[j];
k = Mean[h]; Print[k];
l = sd[h] / Mean[h]; Print[l];
m = Uniformity[h];
Print[m]
```

```
{{-8, -8}, {-8, -4}, {-8, 0}, {-8, 4}, {-8, 8}, {-4, -8}, {-4, -4},
 {-4, 0}, {-4, 4}, {-4, 8}, {0, -8}, {0, -4}, {0, 0}, {0, 4}, {0, 8}, {4, -8},
 {4, -4}, {4, 0}, {4, 4}, {4, 8}, {8, -8}, {8, -4}, {8, 0}, {8, 4}, {8, 8}}
```

```
0.000309908
```

```
{1.43552, 0.339168, 0.210609, 0.339168, 1.43552, 0.339168, 0.0801346, 0.0497603, 0.0801346,
 0.339168, 0.210609, 0.0497603, 0.030899, 0.0497603, 0.210609, 0.339168, 0.0801346,
 0.0497603, 0.0801346, 0.339168, 1.43552, 0.339168, 0.210609, 0.339168, 1.43552}
```

```
0.393933
```

```
0.478605
```

```
0.393933
```

```
1.21494
```

```
0.0421422
```