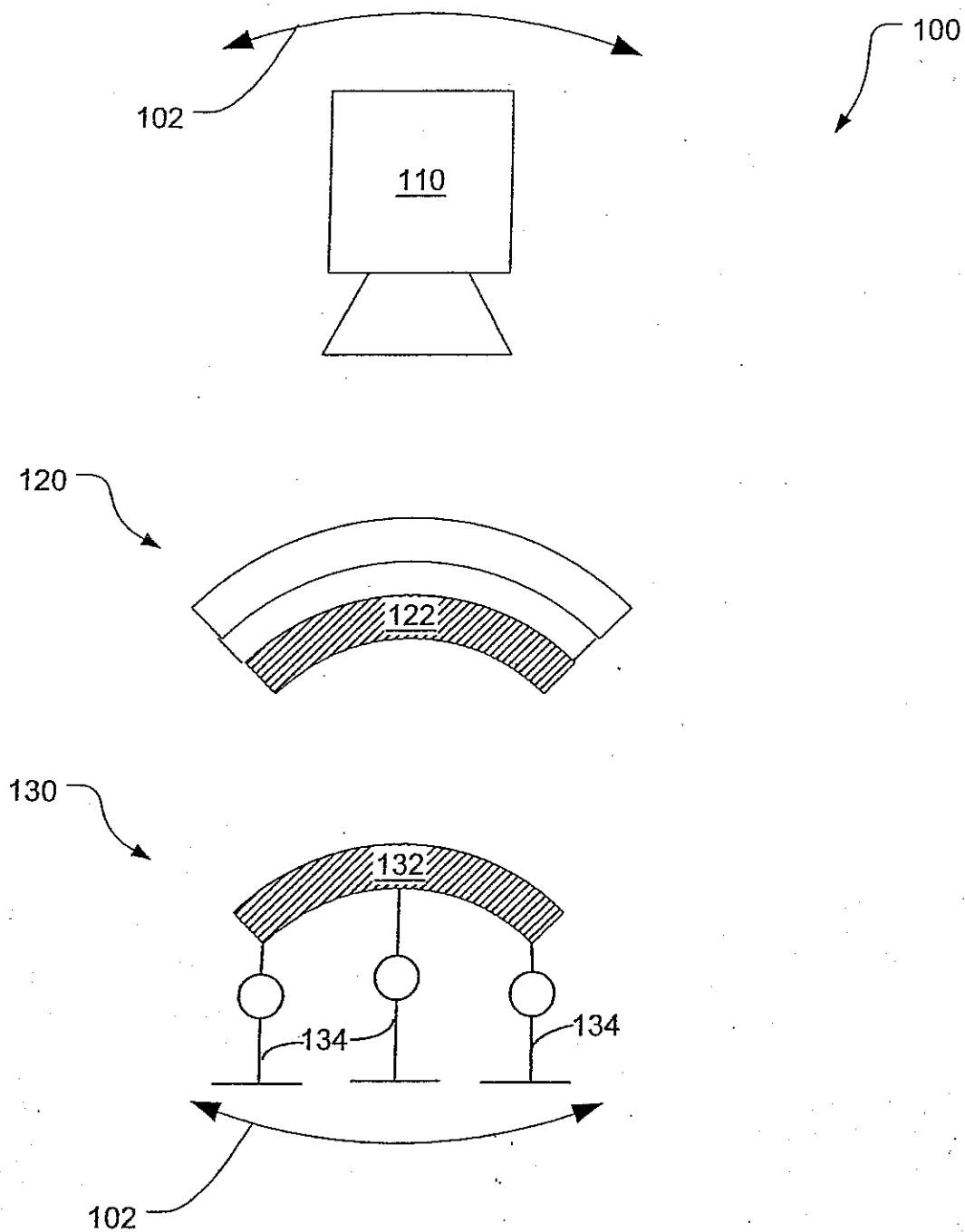
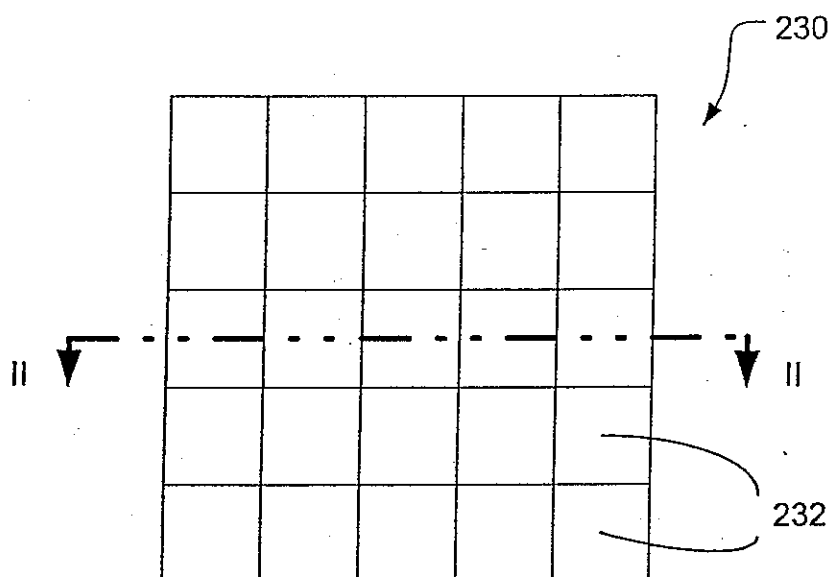
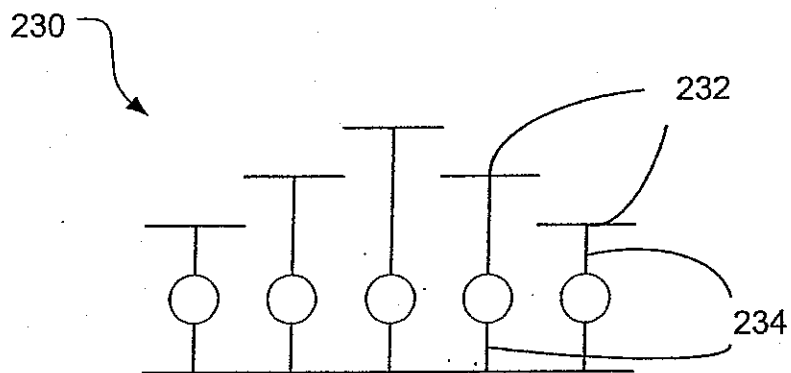


1/12

**FIG. 1**

2/12

**FIG. 2A****FIG. 2B**

3/12

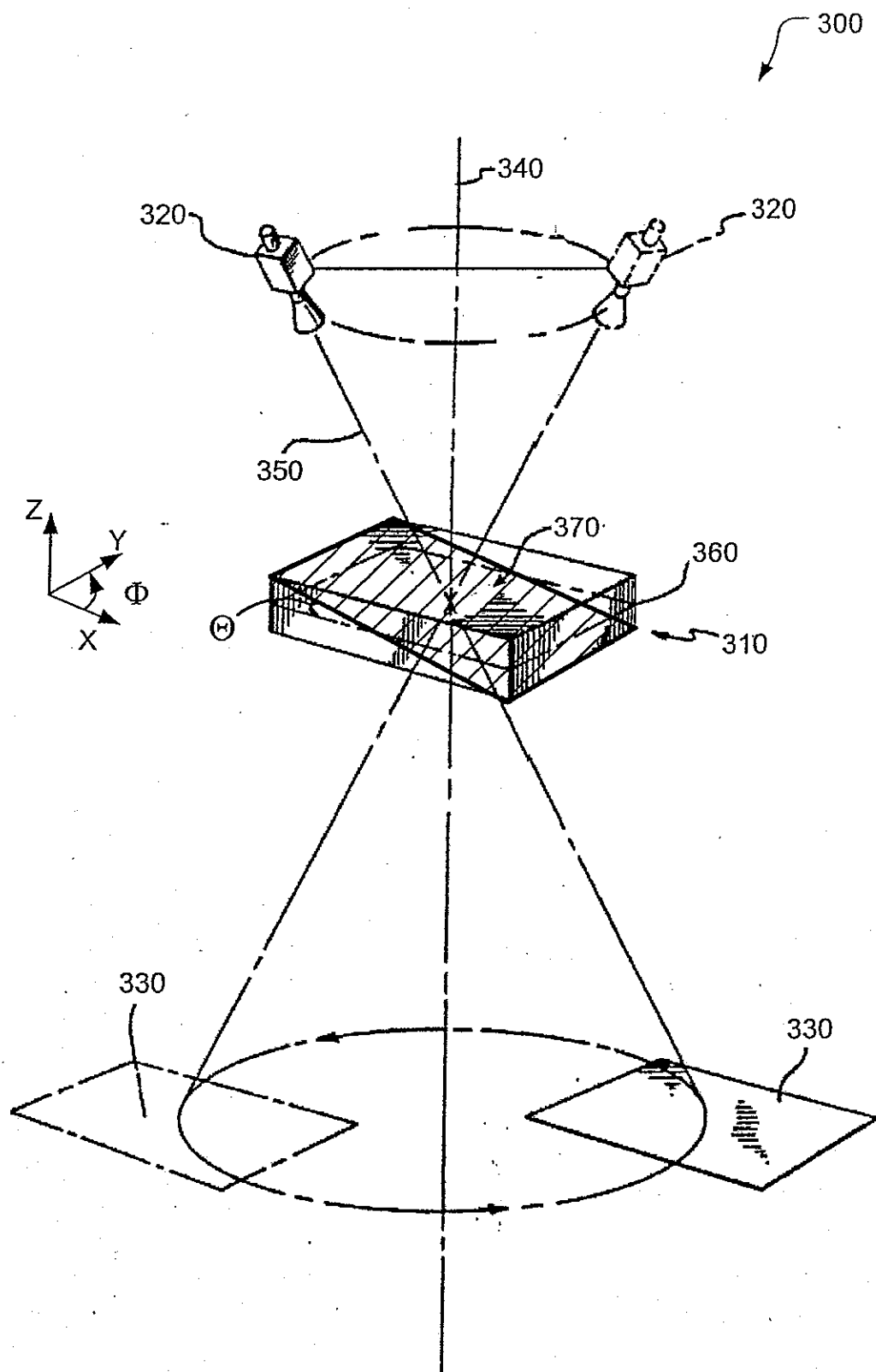


FIG. 3

4/12

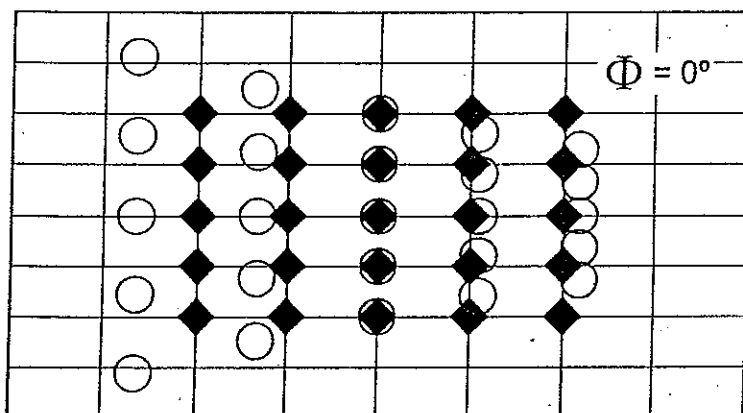


FIG. 4A

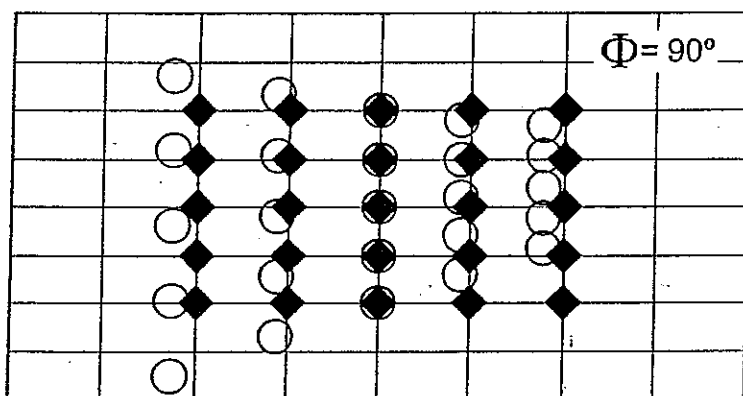


FIG. 4B

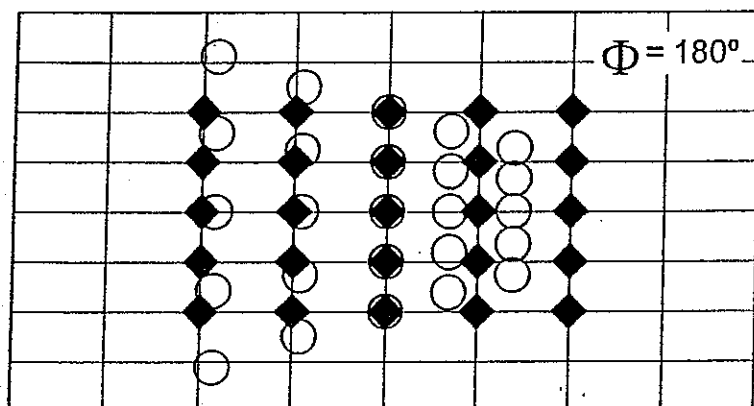


FIG. 4C

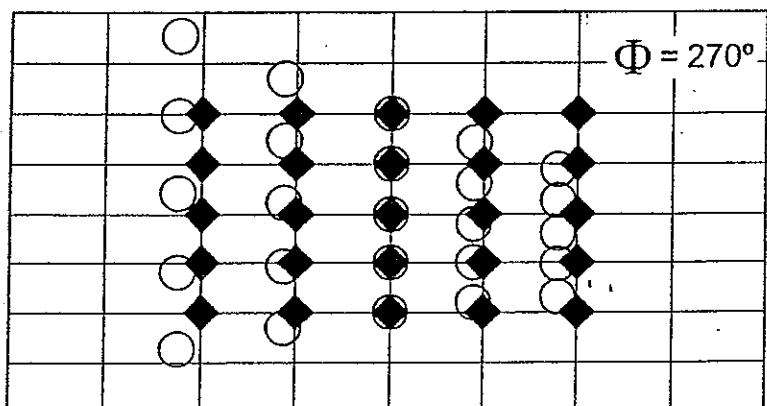


FIG. 4D

5/12

500

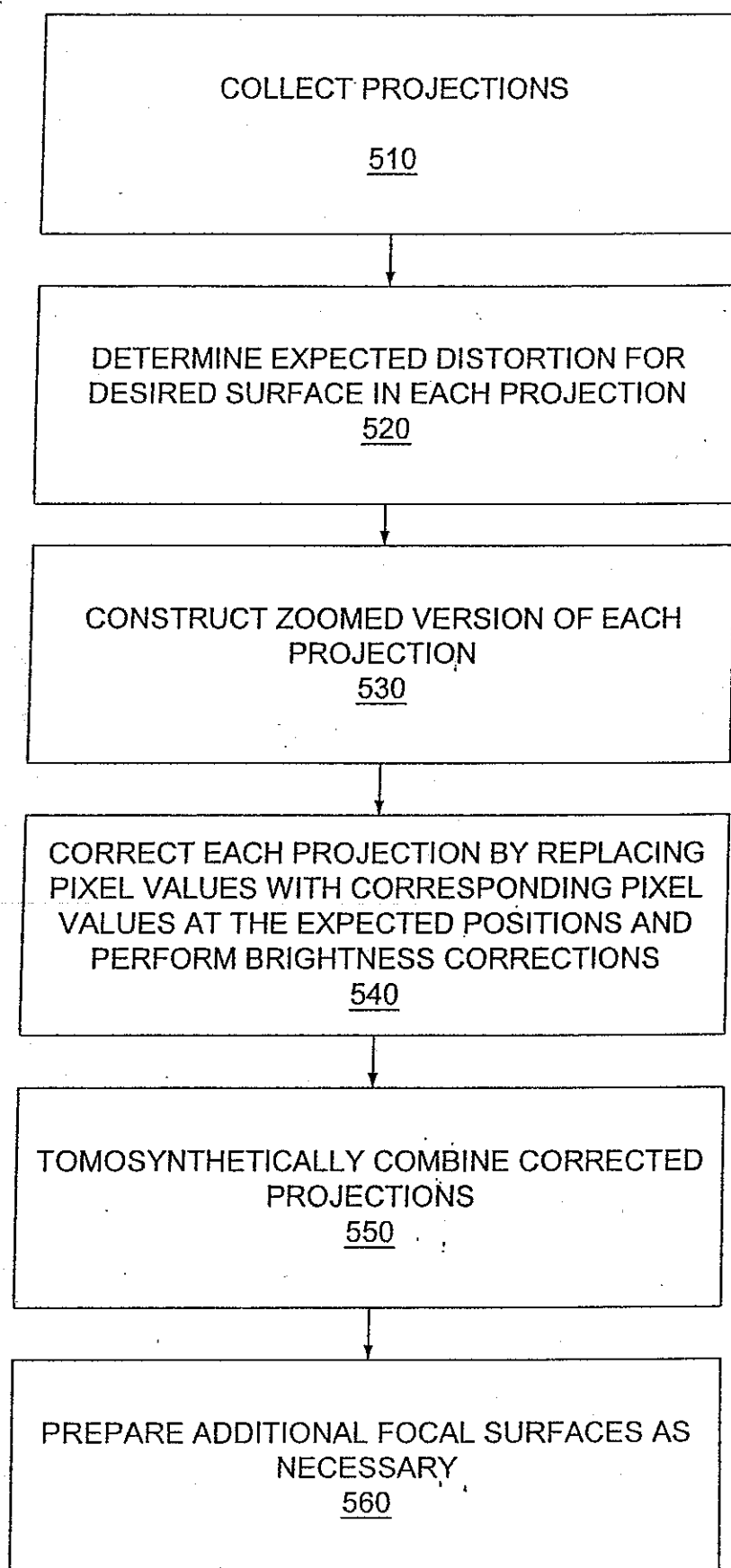
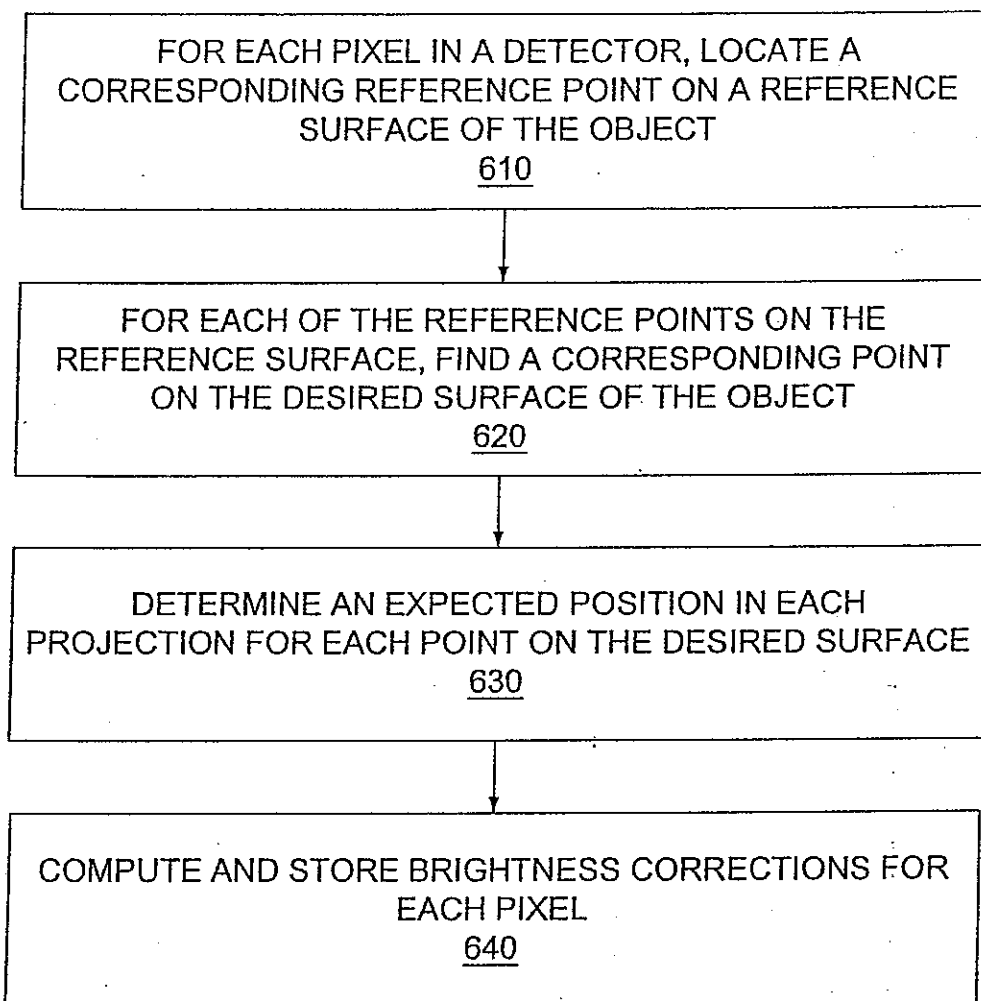


FIG. 5

6/12

520

**FIG. 6**

7/12

560

DETERMINE OFFSETS FOR EACH CORRECTED
PROJECTION IMAGE IN ORDER TO ACHIEVE THE
DESIRED CHANGE IN FOCAL HEIGHT

710

ADJUST MAGNIFICATION OF EACH CORRECTED
PROJECTION SO THAT PIXEL SIZE MATCHES
PIXEL SIZE AT DESIRED FOCAL HEIGHT

720

TOMOSYNTHETICALLY COMBINE CORRECTED
IMAGES USING OFFSETS

730**FIG. 7**

8/12

```

pro warp

n = 256;          N MUST BE EVEN, >= 16

5  while (!d.window ne -1) do wdelete;

    ; define warped surface at origin
    step = fix(n/16);
    zo = shift(dist(n, n), n/2, n/2) * 8 * !pi / (n * sqrt(2));
10  zo = 25*sin(zo)/zo;
    window, 0, xsize=n, ysize=n, title='Warped Surface'
    shade_surf, zo;

    xo = dblarr(n, n);
15  yo = dblarr(n, n);
    obj = bytarr(n, n);
    for i = 0, n - 1 do begin
        xo[i, *] = i - n/2 + 0.5;
        yo[*, i] = i - n/2 + 0.5;
20  endfor;
    for i = step, n - step, step do begin
        obj[i, step:n-step] = 255;
        obj[step:n-step, i] = 255;
    endfor;
25  ndx = where(obj ne 0);    Remember non-zero positions

    ; mesh representation of original surface
    zm=dblarr(n/16, n/16);
    xm=dblarr(n/16, n/16);
30  ym=dblarr(n/16, n/16);
    for i = 0, n / 16 - 1 do begin
        for j = 0, n / 16 - 1 do begin
            xm[i, j] = xo[16 * i, 16 * j];
            ym[i, j] = yo[16 * i, 16 * j];
35  zm[i, j] = zo[16 * i, 16 * j];
        endfor;
    endfor;
    window, 1, xsize=n, ysize=n, title='Warped Surface'
    surface, zm, xm, ym;

40  ; display the original object
    window, 2, xsize=n, ysize=n, title='Flat Object'
    tv, obj;

45  ; define source position
    zs = 1000;
    xs = 707;
    ys = 707;

50  ; define flat detector pixel array
    zd0 = dblarr(n, n) - 9000;
    mag0 = (-9000 - zs) / (-zs);    Flat object to flat detector
    x0 = xs + mag0 * (xo - xs);
    xc0 = xs - mag0 * xs;
55  y0 = ys + mag0 * (yo - ys);
    yc0 = ys - mag0 * ys;
    ix = floor((x0 - xc0) / mag0 + n / '2);

```

FIG. 8A

9/12

```

iy = floor((y0 - yc0) / mag0 + n / 2);
img = bytarr(n, n);
img[ix[ndx], iy[ndx]] = obj[ndx];
window, 3, xsize=n, ysize=n, title='Flat Obj/ Flat Det'
5  tv, img;

; image warped object on flat detector
mag = (zd0 - zs) / (zo - zs); Warped object to flat detector
x1 = xs + mag * (xo - xs);
10 y1 = ys + mag * (yo - ys);
ix = floor((x1 - xc0) / mag0 + n / 2);
iy = floor((y1 - yc0) / mag0 + n / 2);
img = bytarr(n, n);
img[ix[ndx], iy[ndx]] = obj[ndx];
15 window, 4, xsize=n, ysize=n, title='Warped Obj/ Flat Det'
tv, img;

; define warped detector pixel array
zd = zd0 + mag * zo;
20 magw = (zd - zs) / (zo - zs);
x2 = xs + magw * (xo - xs);
y2 = ys + magw * (yo - ys);
ix = floor((x2 - xc0) / mag0 + n / 2);
iy = floor((y2 - yc0) / mag0 + n / 2);
25 img = bytarr(n, n);
img[ix[ndx], iy[ndx]] = obj[ndx];
window, 5, xsize=n, ysize=n, title='Warped Obj/Warped Det'
tv, img;

30 end;

```

FIG. 8B

10/12

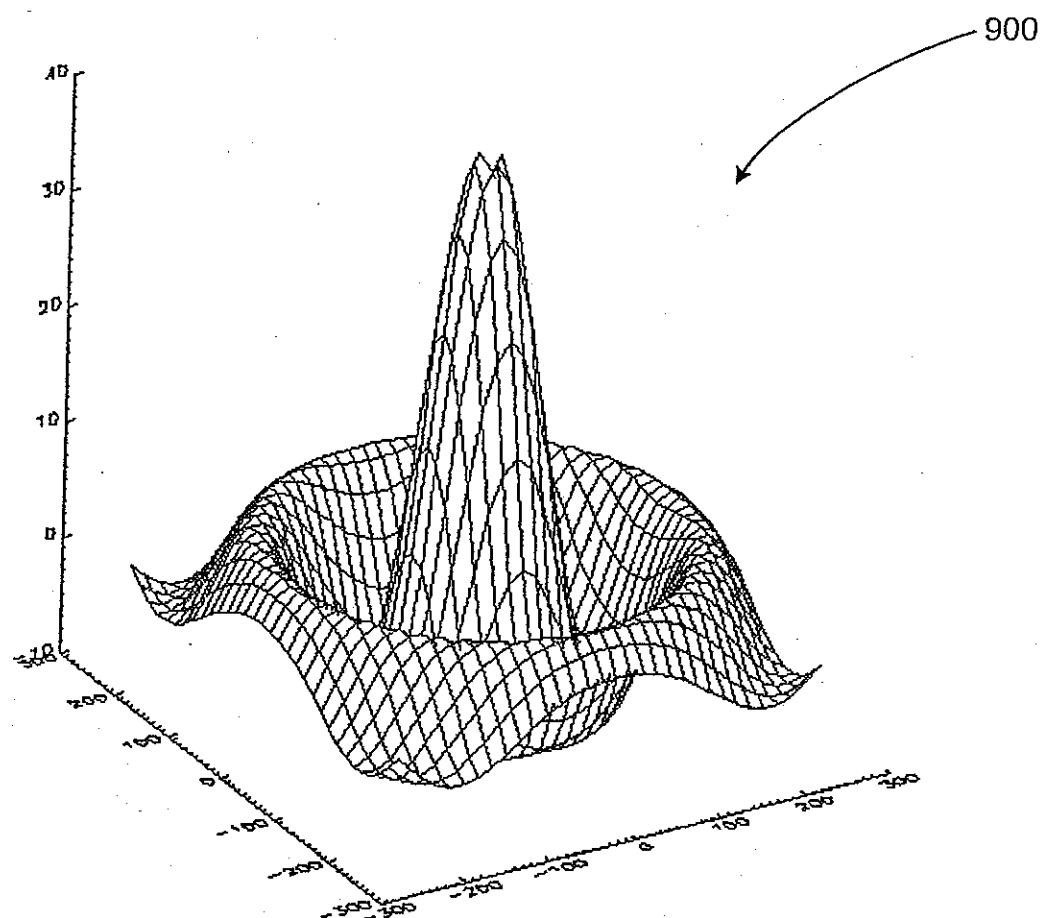


FIG. 9

11/12

1000

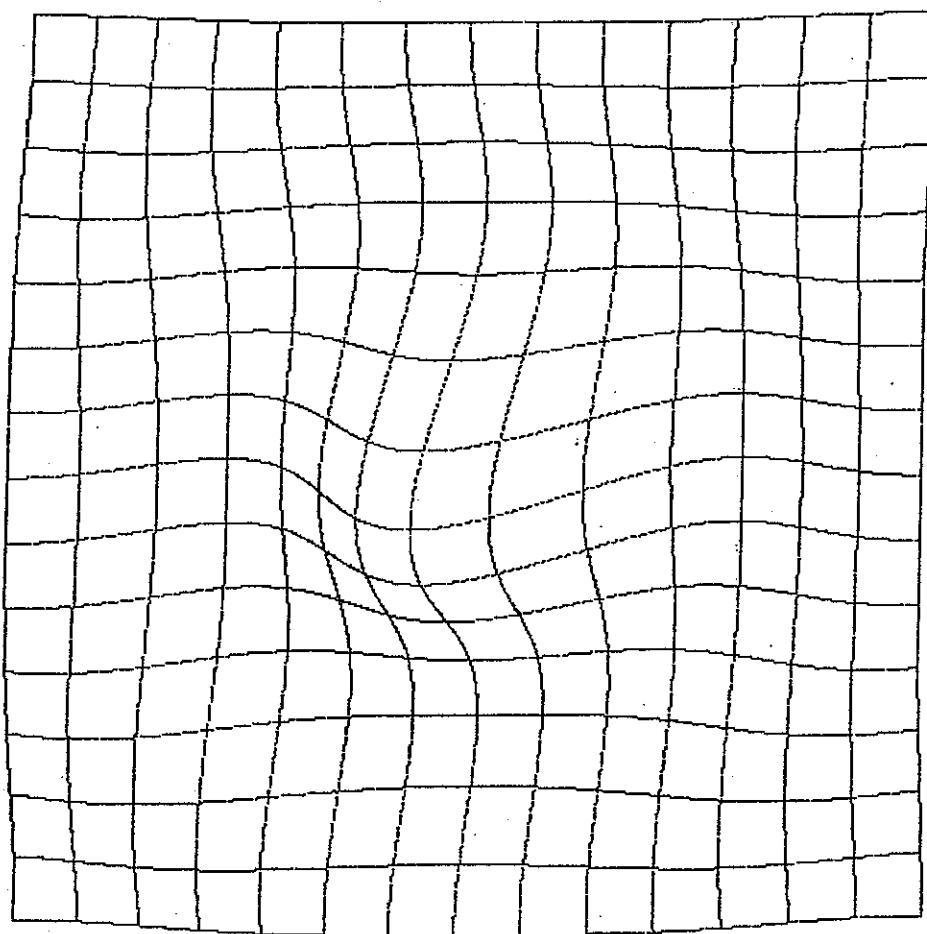


FIG. 10

12/12

1100

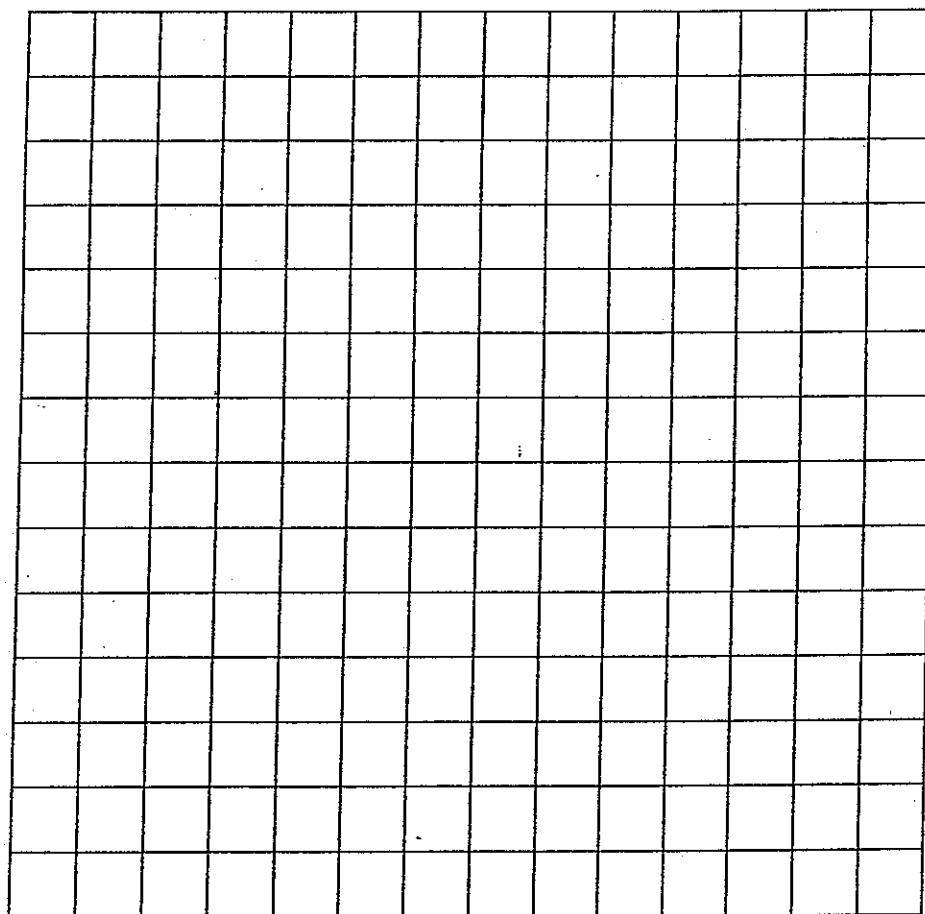


FIG. 11