

2000- 000

2000

「
」

2000. 12. 31.

1 : ()
2 : ()
: ()
: ()
()
()
()

1.

2.

2000 5 1 - 2000 12 31

3.

()
()

4.

가.

		1	2	3	4	5	6	7	8	9	10	11	12
가. 가 · , , 가, · 가 SW tool 가 가 · , · , ·													
(%)						25		65		100			

.

1)

○ ()

.

○ () .

2)

○

○

3)

가

○

○

4)

,

○ .

5.

1)

2)

3)

4)

6.

,

.

.

7.

.			
	Pentium	8	
WorkStation	SW S715/ 100	1	
Measuring Receiver	ML524B	1	
Frequency Converter	MH669B	1	
Signal Generator	HP8664A	1	
Antennas	dipole(MP651B) microstrip	6	

8.

ABSTRACT

A Study on Developing an Indoor Propagation Model in Semi-Microwave Band

In this study, we proposed the image-based 3D ray-tracing indoor propagation model using patch scattering model which can calculate the scattering phenomenon of the indoor structures. An image-based 3D ray-tracing technique allows rapid and exact results because it makes from indoor space to a rectangular parallelepiped equivalently and solves ray paths by generating image antennas about each reflection plane. However it requires a extremely complex solution about arbitrary indoor structures. A patch scattering model for modeling indoor structures defines a scattering phenomenon by using RCS(Radar Cross Section) about rectangular patch without complex calculation. RCS is simply defined as a ratio of scattering power to incident power, and we use bistatic RCS which is simplified numerically by Physical Optics. Also, a simple indoor compensation factor is defined as empirical constant from measured data instead of complex numerical expression because basic patch scattering model cannot include important multipath components.

For application of patch scattering model about arbitrary indoor structures, we consider scattering loss from surface roughness of patch. If patch has lower roughness height then Rayleigh threshold, it can be assumed as perfect plane. If not, patch scattering model is calculated by using scattering loss from standard deviation about roughness of patch.

To prove proposed indoor propagation model using patch scattering model, comparison cumulative curve of signal and signal envelope between simulation and measured data was shown, and this comparison provides close match. So, patch scattering model reflects scattering phenomenon of patch well.

Hence, we can consider this proposed model has good application about indoor environment including arbitrary structures and be also used for the design of the wireless networks in arbitrary indoor space.

	10
	11
1	17
2	3 21
1	 24
2	 30
3	 44
3	 65
1	RCS 65
2	 68
4	 87
1	 87
2	 94
3	 101
5	 111
	115
	119

2- 1	(@1890MHz)	27
2- 2	59
2- 3		
	59
3- 1	79
4- 1	89
4- 2	95

2- 1	22
2- 2	23
2- 3	23
2- 4 LOS	25
2- 5 NLOS	26
2- 6	28
2- 7	29
2- 8 MSA	29
2- 9 1	32
2- 10 가	33
2- 11 가	34
2- 12	35
2- 13	37
2- 14	38
2- 15	가	40
2- 16	41
2- 17	43
2- 18	44
2- 19	45
2- 20 1GHz	46
2- 21 2GHz	46
2- 22	47
2- 23	48
2- 24	49
2- 25	51
2- 26	53

2-27		54
2-28		54
2-29		58
2-30		61
2-31		DOA AOS	
		61
3-1	RCS	66
3-2	RCS	67
3-3	a RCS	68
3-4		69
3-5		69
3-6		72
3-7		74
3-8		74
3-9		75
3-10		76
3-11		77
3-12		77
3-13			78
3-14		78
3-15		80
3-16	1	(:1m × 1m).....	81
3-17	9	(: 1m × 1m, : 3m).	81
3-18	1	(:0.2m × 0.2m).....	82
3-19	9	(:0.2m × 0.2m, : 3m).	82
3-20	9	(: 1m × 1m, :	

	1.1m).	83
3- 21	9 (:1m × 1m, :7 m).	83
4- 1		88
4- 2		90
4- 3		91
4- 4		92
4- 5		92
4- 6		93
4- 7		93
4- 8		94
4- 9		95
4- 10		96
4- 11		96
4- 12	(1).	97
4- 13	(2).	98
4- 14	(3).	99
4- 15	(4).	100
4- 16	1	102
4- 17	2	103
4- 18	3	107
4- 19	4	107

1

1

,
.
(ray tracing) 가
.
가
, (geometrical optics)
UTD(Uniform geometrical Theory of Diffraction)
[1]. 가 가
(ray launching) ,
UTD 가 .
,
가
.
가
가 .
(image method)
.
가 .
가
.
[2,3,4]
,

가 .

.

.

RCS(Radar Cross Section) .

.

가 .

2 3 ,

3 .

4

5 .

2 3

2 3

.

.

, , , ,
.

가

.

[3,4,5]

.

,

,

,

.

,

.

.

3

3

가

,

UTD

.

.

가

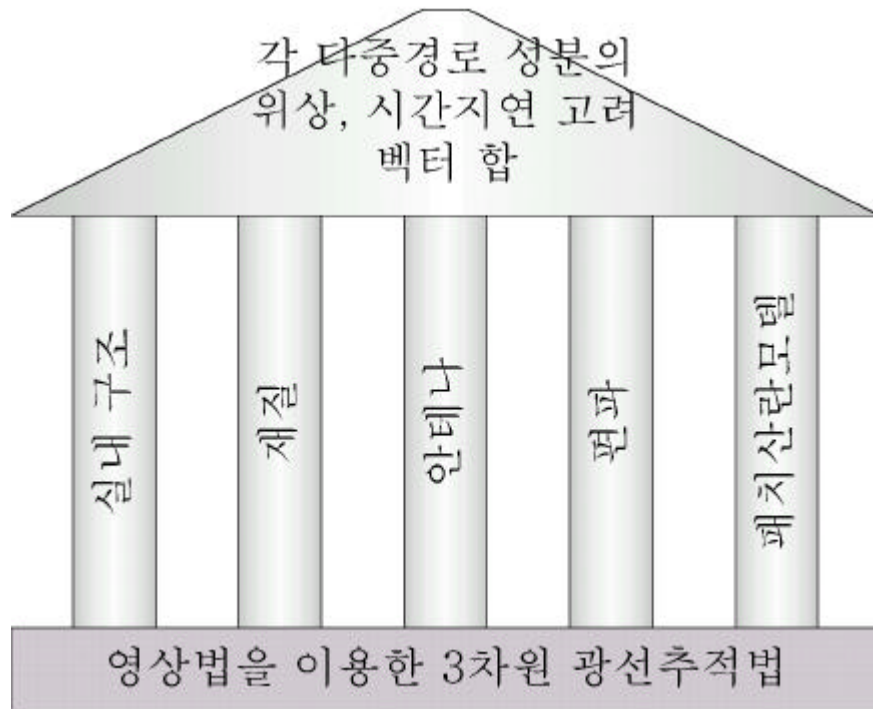
. ,

. ,

,

.

,



2-7.

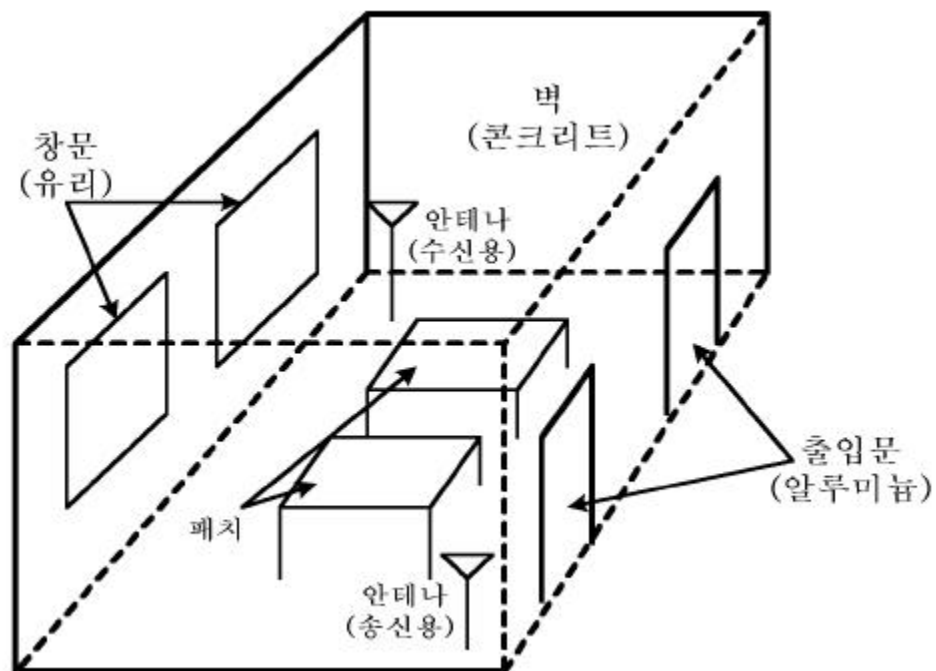
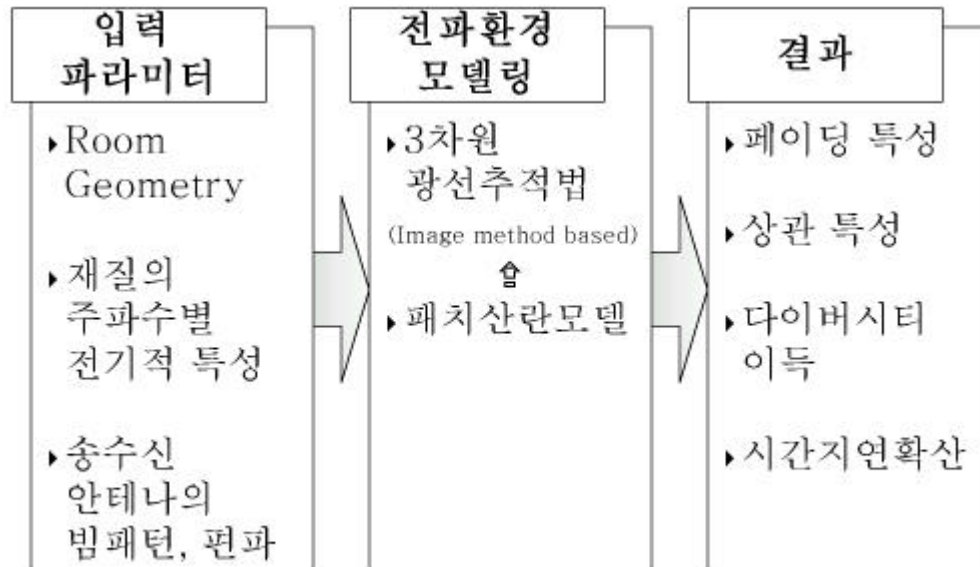
2- 2

가

2-3

3

가

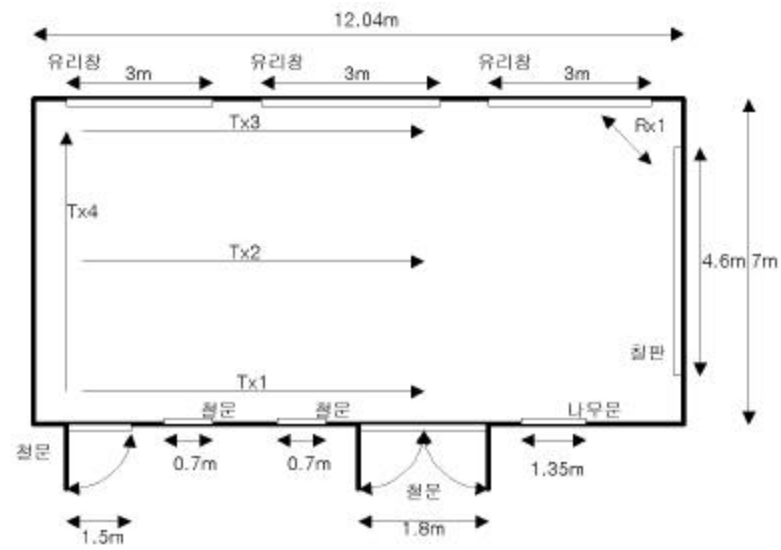


1

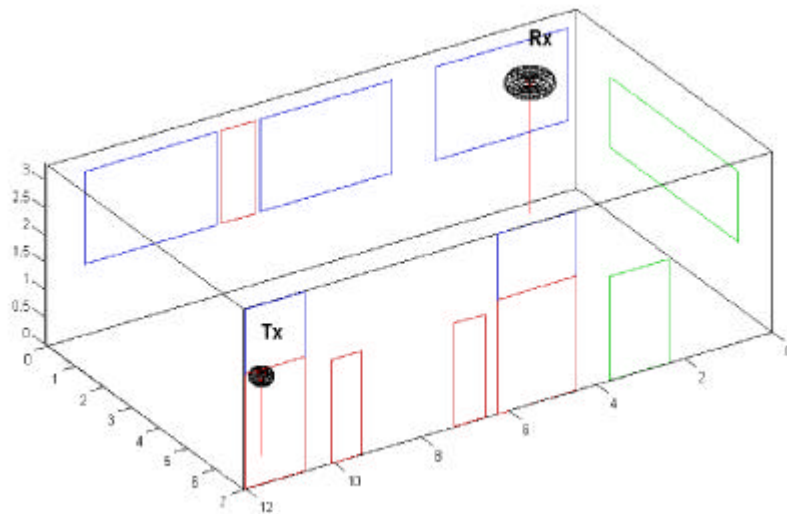
1.

가 (NLOS) , 가 (LOS) 가 ‘ γ ’

2-4 2-5

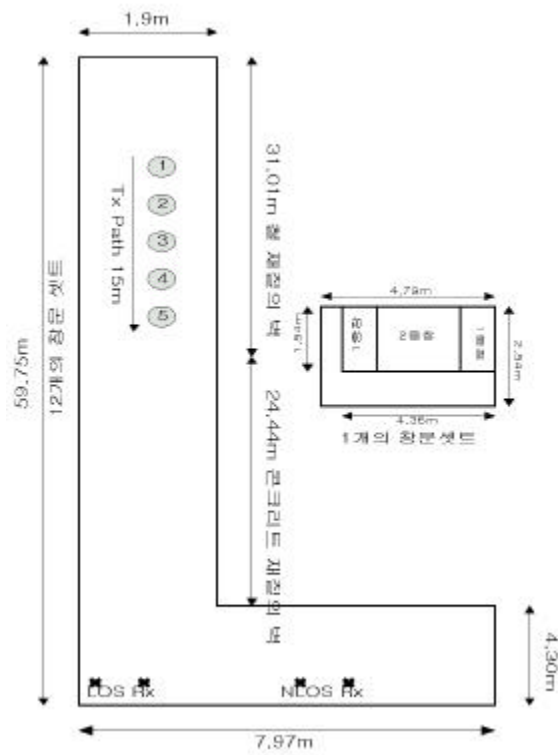


(a) LOS

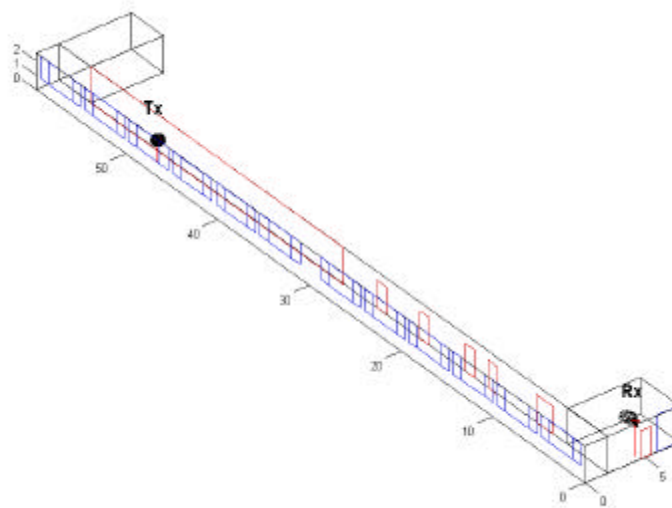


(b)

2-4. LOS



(a) NLOS



2-5. NLOS

2.

가

가

2- 1

[4]

가

가

가

2- 6

2.5, 3, 3.5

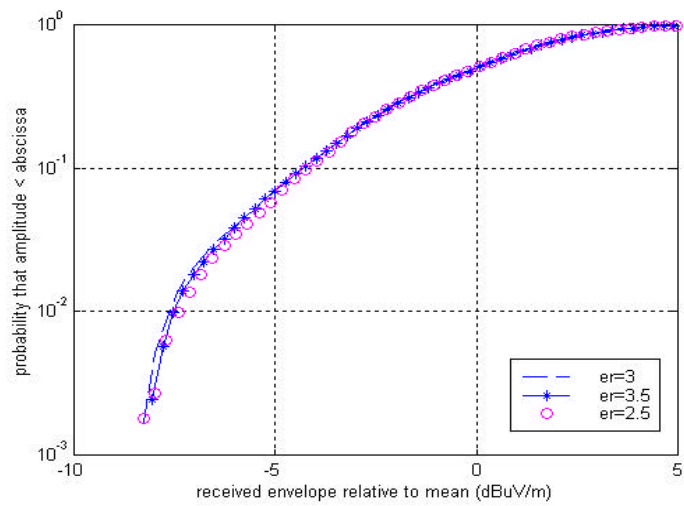
. 1

가

2- 1.

(@1890MHz)

	[S]		[m] <i>h</i>
half height windows	5×10^{-4}	3	2×10^{-3}
solid reinforced concrete	5×10^{-3}	2.7	2×10^{-3}
plasterboard & steel frame	5×10^{-2}	3	1×10^{-3}
plasterboard & timber frame	5×10^{-3}	2.7	1×10^{-3}
metal	9×10^6	1	0
timber door	3×10^{-4}	2	1×10^{-3}



2-6.

3.

[6]

가

1×4 , 4×4

(MSA:MicroStrip Array antenna) .

가 0° , 45° , 90° .

(unit vector)

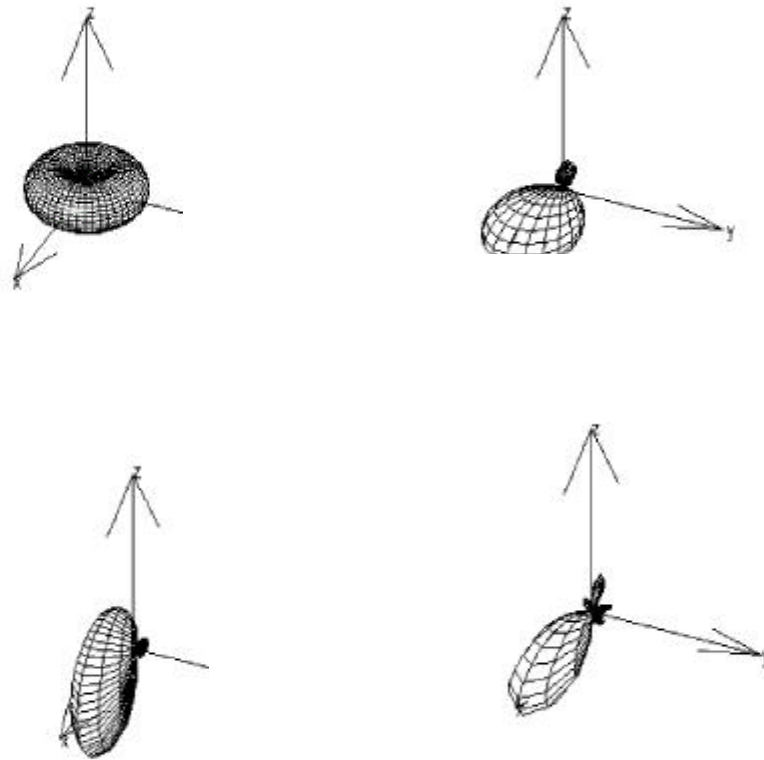
2-7

dipole, 1×4 vertical array MSA, 1×4 horizontal array MSA

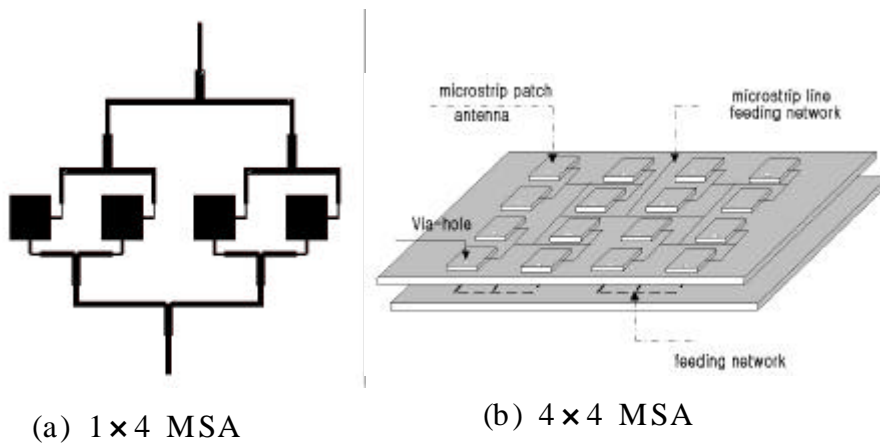
4×4 MSA

2-8

MSA



2- 7.



2- 8. MSA

2

3

가

가

3

1. 3

가.

(2- 1)

$$h(t)=\sum_{n=1}^NA_n\delta(t-\tau_n)e^{-j\theta_n}$$

(2- 1)

Dirac

$h(t)$

(2- 2)

$$V_i=\frac{G_T}{G_R}\frac{e^{-j\beta r_d}}{r_d}R_{loss}D_{loss}\Phi_N\cdot\Phi_R$$

(2- 2)

G_T G_R 3

r_d , R_{loss} D_{loss}

$$,\quad \Phi_N\quad \Phi_R$$

$$\cdot\quad (2-2)$$

·

·

·

·

가

·

(cavity)

,

,

,

·

가

가

, 가

, 가

·

가

,

가

UTD

·

,

- -

·

,

·

2-9

1

1

·

2

1

18

,

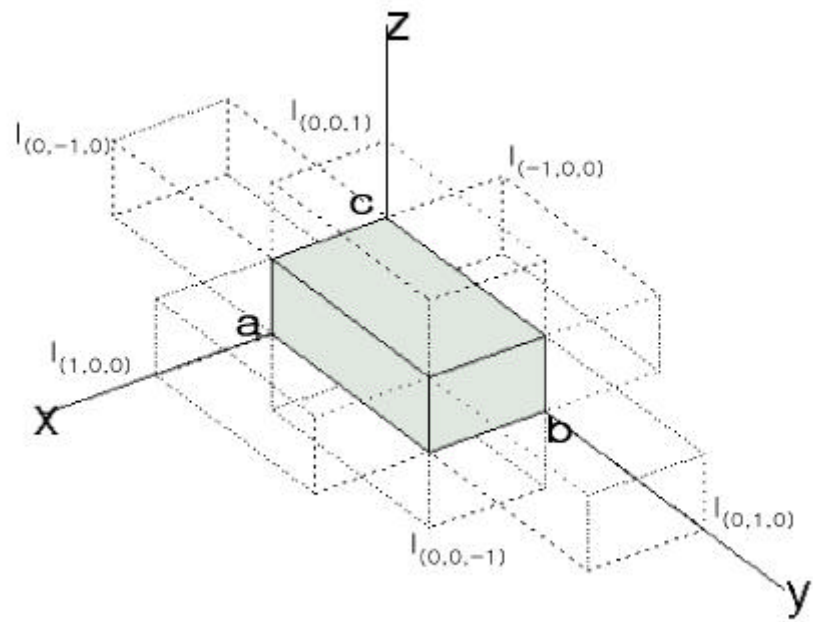
가

·

, N

$$I_N = 4N^2 + 2 \quad , \quad 1 \quad N$$

$$\sum_{s=0}^N (4s^2 + 2) = \frac{3}{4} N^3 + 2N^2 + \frac{8}{3} N \quad .$$



2-9. 1

가

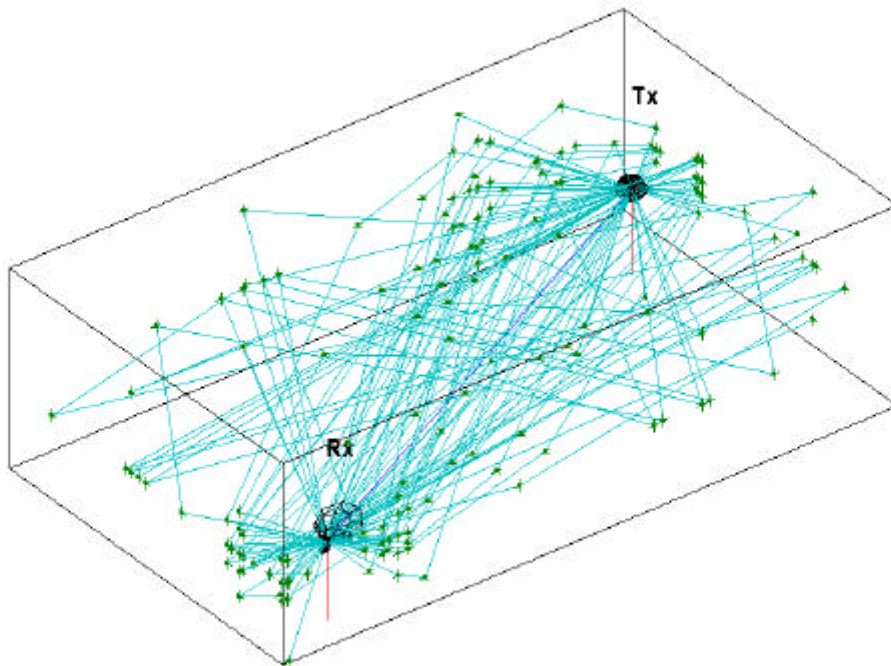
2- 10 가

2- 11 가

2- 10 (N)가 3 , 62 (R)

가
가 ,

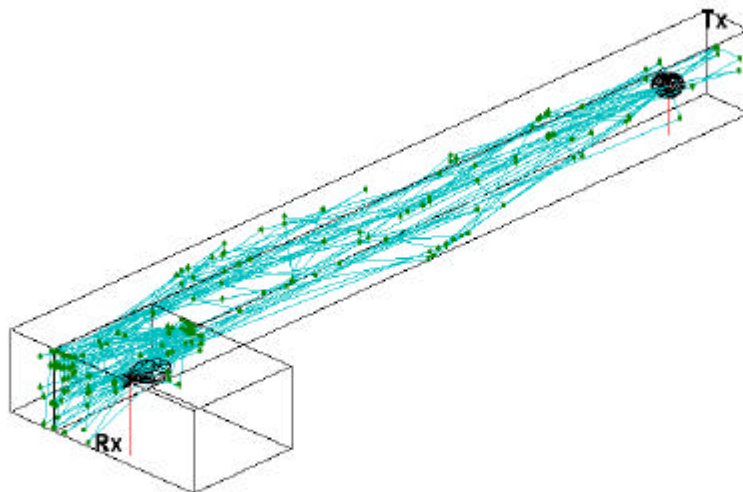
3



2- 10. 가

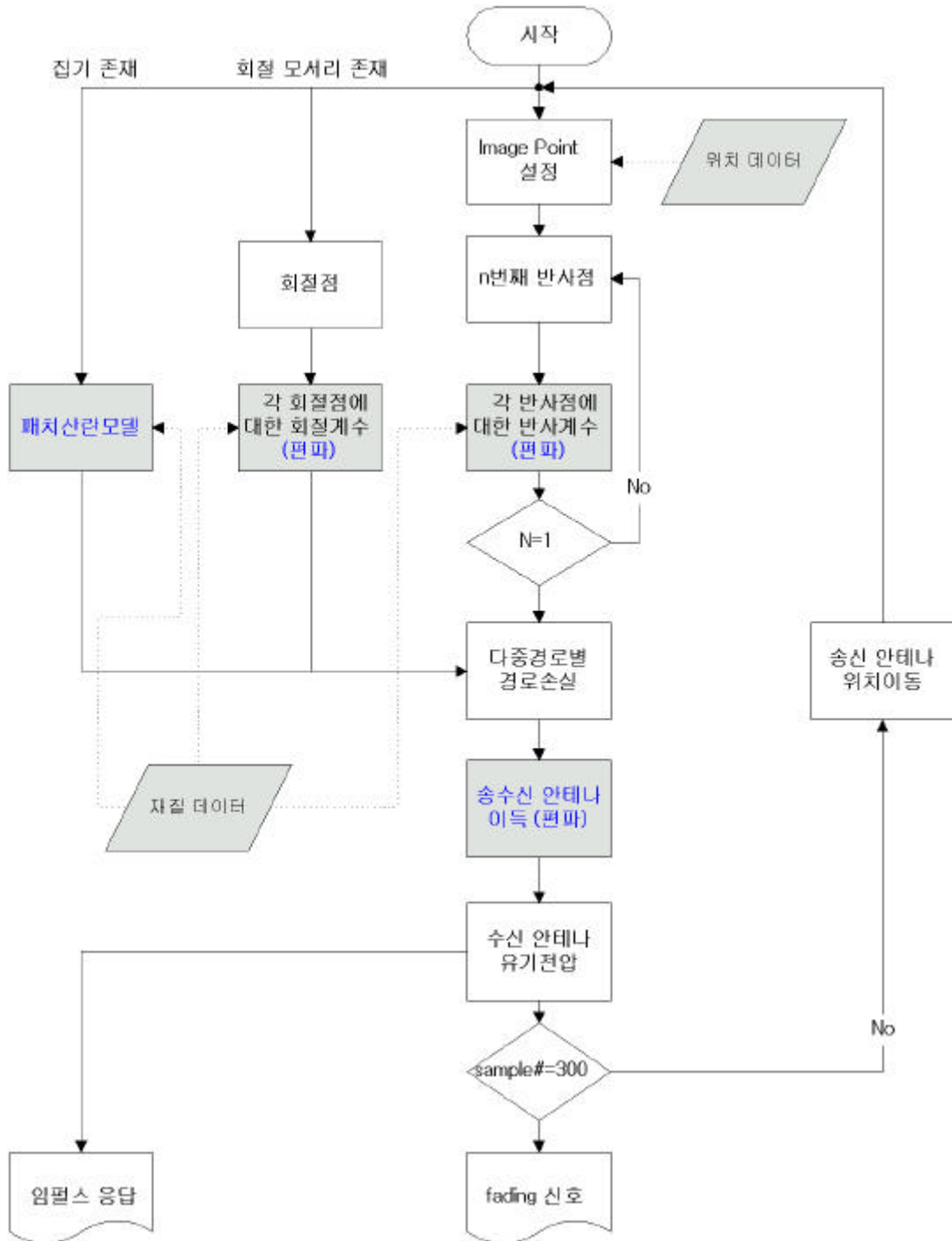
(N=3, R=62)

2-11 가 , 가 , 가 , 가 . 2-11 가 4 , 29 .



2-11. 가

(N=4, R=29)



2.

,
 ,
 ,
 ,
 ,
 .

가.

UTD

,
 .
 ,

(fresnel coefficient) ^[7] .

,
(2-3) .

$$\Gamma_{\parallel}(\theta_i)=\frac{\sqrt{\epsilon_r-\sin^2\theta_i}-\epsilon_r\cos\theta_i}{\epsilon_r\cos\theta_i+\sqrt{\epsilon_r-\sin^2\theta_i}}$$

(2-3)

$$\Gamma_{\perp}(\theta_i)=\frac{\cos\theta_i-\sqrt{\epsilon_r-\sin^2\theta_i}}{\cos\theta_i+\sqrt{\epsilon_r-\sin^2\theta_i}}$$

_{*i*} 2-13 가
 , _{*r*} .
(2-4) (ϵ_r') (),

$$\epsilon_r = \epsilon'_r - j \frac{\sigma}{\omega \epsilon_0} \quad (2-4)$$

ω : , ϵ_0 :

2- 1

2- 14 . 2- 14(a)

가 가

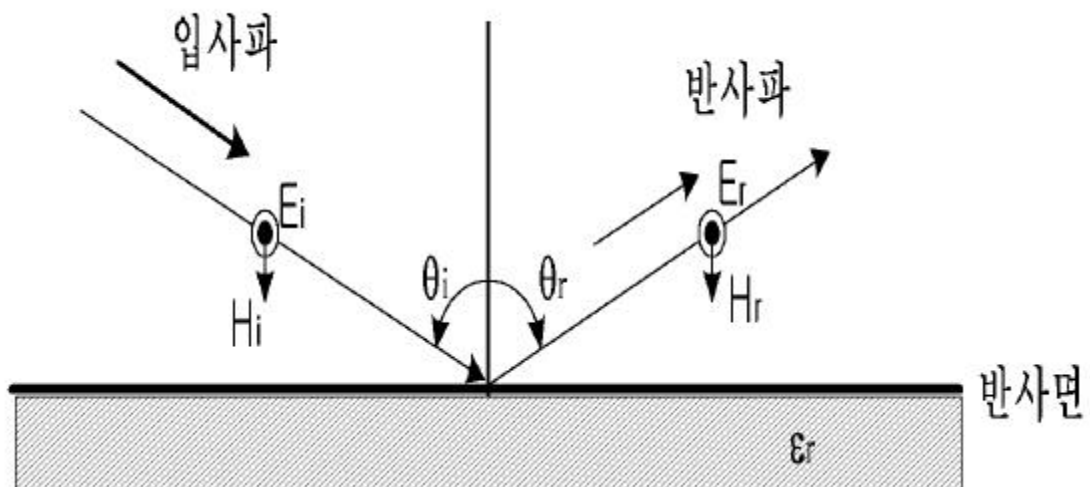
(Brewster angle)

가 가

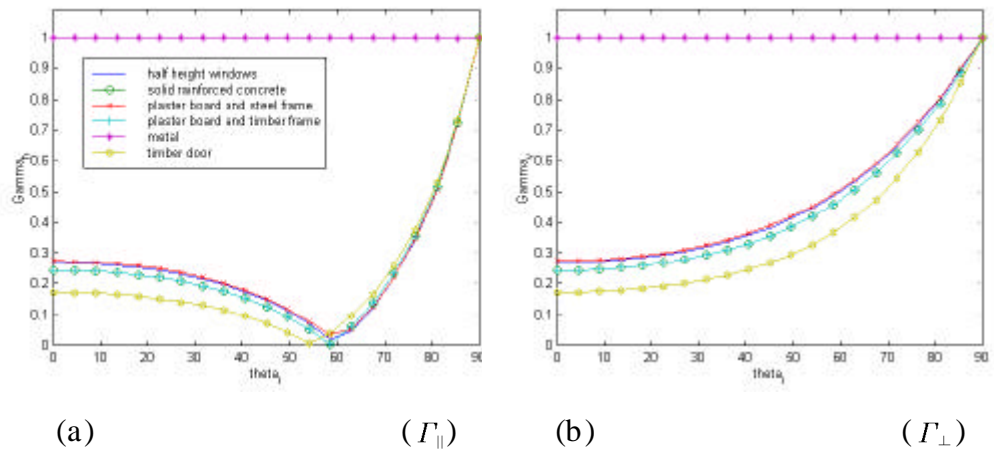
0 가 90

가 1 . 2- 14(b)

가 가 .



2- 13.



2- 14.

.

(2- 3) (smooth) 가

.

, (rough)

.

, ,

, (2- 5) (2- 7)

.

$$h_c = \frac{\lambda}{8 \sin \theta_i} \tag{2- 5}$$

h_c , 가

,

.

,

.

$$\rho_s = I_0 [8(\frac{\pi \sigma_h \sin \theta_i}{\lambda})^2] \cdot \exp [- 8(\frac{\pi \sigma_h \sin \theta_i}{\lambda})^2] \tag{2- 6}$$

$$\sigma_n \qquad \qquad \qquad , I_o[\cdot] \qquad \qquad \qquad 0$$

$$\qquad \qquad \qquad , \rho_s \qquad \qquad \qquad ,$$

$$F_{rough} \qquad \qquad \qquad .$$

$$F_{rough} = \rho_s F \qquad \qquad \qquad (2-7)$$

.

,

$$\text{가} \qquad \qquad \qquad [16] .$$

$$\text{가} \qquad \qquad \qquad , \qquad \qquad \qquad ,$$

.

$$\text{UTD} \qquad \qquad \qquad [1] . \qquad \qquad \qquad \text{Keller가}$$

$$\text{가} \qquad \qquad \qquad \text{가} \qquad \qquad \qquad \text{가}$$

$$\text{Kouyoumjian} \quad \text{Pathak} \qquad \qquad \qquad . \qquad \qquad \qquad , \text{Leubbers}$$

$$\text{가} \qquad \qquad \qquad \text{GTD} \qquad \qquad \qquad [15] .$$

$$\text{가} \qquad \qquad \qquad , \qquad \qquad \qquad \text{가 } A(s',s) \text{ (spreading divergence factor)} \\ \qquad \qquad \qquad . A(s',s) \qquad \qquad \qquad (2-8)$$

$$\qquad \qquad \qquad s' \qquad \qquad \qquad s$$

$$\qquad \qquad \qquad 2-15 \qquad \qquad \qquad \text{가}$$

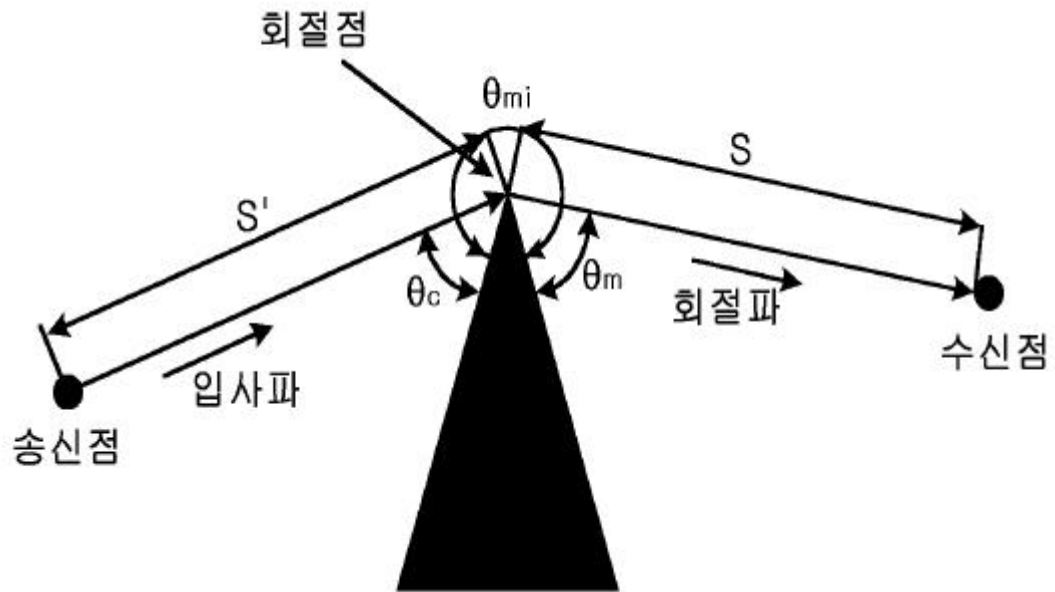
.

$$A(s',s) = \sqrt{\frac{s'}{s(s'+s)}} \qquad \qquad \qquad (2-8)$$

$$(2-9) \qquad \qquad \qquad Q_d \qquad \qquad \qquad .$$

$$E^d(s) = E^i(Q_d) \overline{DA}(s',s) e^{-j\beta s} \qquad \qquad \qquad (2-9)$$

가 D 가 , 가



2- 15. 가

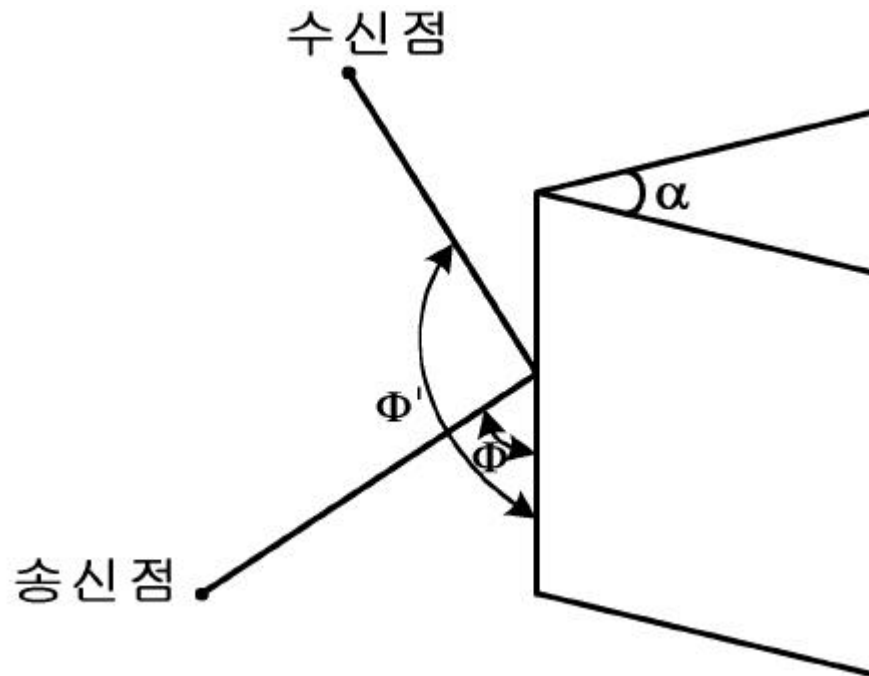
$$D_{\parallel}^S = \frac{\exp\left(-\frac{j\pi}{4}\right) \frac{1}{n} \sin\left(\frac{\pi}{n}\right)}{\sqrt{2\pi\beta}} \quad (2-10)$$

$$\times \left[\frac{1}{\cos\left(\frac{\pi}{n}\right) - \cos\left(\frac{\phi - \phi'}{n}\right)} - \frac{1}{\cos\left(\frac{\pi}{n}\right) - \cos\left(\frac{\phi + \phi'}{n}\right)} \right]$$

$$(2-10) \quad D_{\parallel}^S \quad \text{가} \quad (2-11)$$

$$D_{\perp}^S$$

ϕ , ϕ' , n 가
 $n = (2\pi - \alpha)/\pi$. (2-12) (2-13) 가
 가 ,



2- 16.

$$\begin{aligned}
D_{\parallel}^s = & \frac{e^{-j\pi/4}}{2n\sqrt{2\pi k \sin \beta}} \left\{ \cot\left(\frac{\pi + (\phi - \phi')}{2n}\right) F(kL a^+(\phi - \phi')) \right. \\
& + \cot\left(\frac{\pi - (\phi - \phi')}{2n}\right) F(hL a^-(\phi - \phi')) \\
& + \Gamma_{\parallel} \left(\cot\left(\frac{\pi + (\phi + \phi')}{2n}\right) F(kL a^+(\phi + \phi')) \right. \\
& \left. \left. + \cot\left(\frac{\pi - (\phi + \phi')}{2n}\right) F(kL a^-(\phi + \phi')) \right) \right\}
\end{aligned} \tag{2- 12}$$

$$\begin{aligned}
D_{\perp}^s = & \frac{e^{-j\pi/4}}{2n\sqrt{2\pi k \sin \beta}} \left\{ \cot\left(\frac{\pi + (\phi - \phi')}{2n}\right) F(kL a^+(\phi - \phi')) \right. \\
& + \cot\left(\frac{\pi - (\phi - \phi')}{2n}\right) F(hL a^-(\phi - \phi')) \\
& + \Gamma_{\perp} \left(\cot\left(\frac{\pi + (\phi + \phi')}{2n}\right) F(kL a^+(\phi + \phi')) \right. \\
& \left. \left. + \cot\left(\frac{\pi - (\phi + \phi')}{2n}\right) F(kL a^-(\phi + \phi')) \right) \right\}
\end{aligned} \tag{2- 13}$$

$$, \quad F(x) \quad .$$

$$F(x) = 2j\sqrt{x} e^{jx} \int_{\sqrt{x}}^{\infty} e^{(-j)\tau^2} d\tau \tag{2- 14}$$

$$L = \frac{ss'}{s + s'} \tag{2- 15}$$

$$a^{\pm}(\beta) = 2\cos^2\left(\frac{2n\pi N^{\pm} - \beta}{2}\right), \quad \beta = \phi \pm \phi' \tag{2- 16}$$

$$, \quad N \quad .$$

$$2\pi nN^+ - \beta = \pi \tag{2- 17}$$

$$2\pi nN^- - \beta = -\pi$$

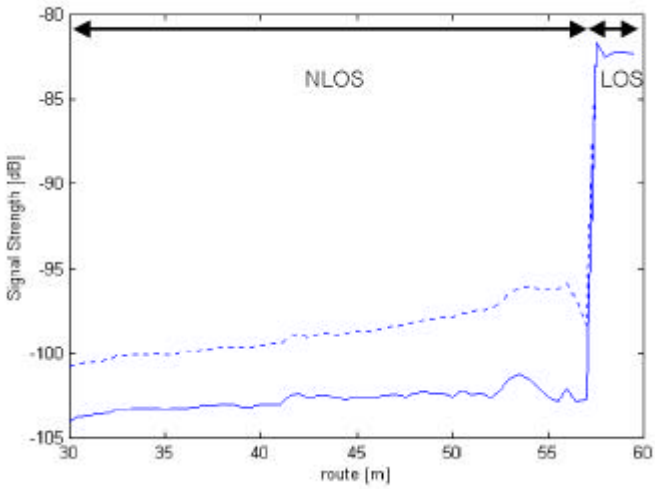
(2- 12)
(2- 13)

. (2- 18) .

$$\begin{bmatrix} E_{\parallel}^s \\ E_{\perp}^s \end{bmatrix} = \begin{bmatrix} -D_{\parallel}^s & 0 \\ 0 & -D_{\perp}^s \end{bmatrix} \begin{bmatrix} E_{\parallel}^i \\ E_{\perp}^i \end{bmatrix} \tag{2- 18}$$

2- 17 가 가 가 ,

. GO
UTD . UTD 가
가 .



2- 17.
(:GO, :GO+UTD)

3

1.

가

Rayleigh, Rician

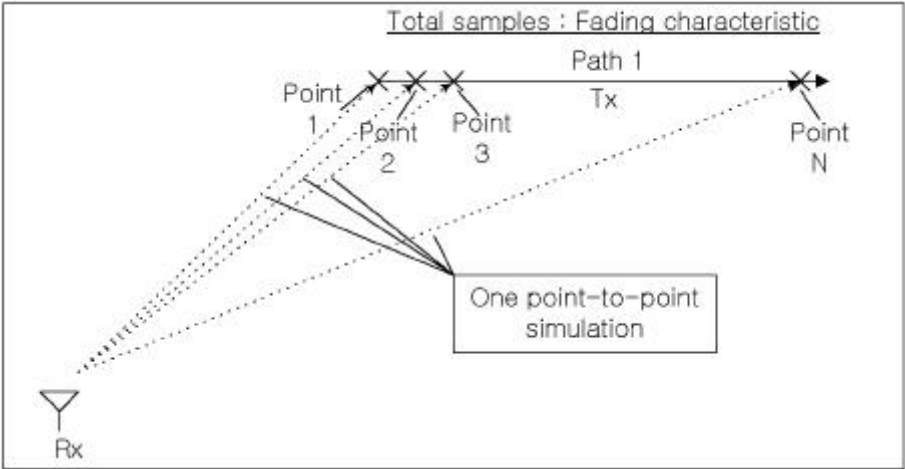
Rician

X 10dB

가

2- 18

N



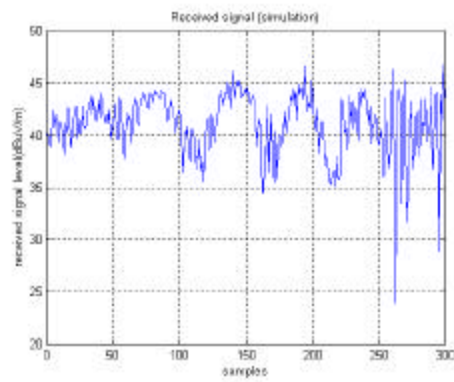
2- 18.

2- 19

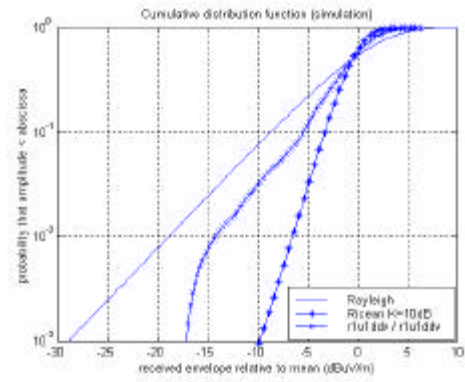
2- 19(a)

, 2- 19(b)
가 .

. 2- 19(b)
LOS Rician K



(a)



(b)

2- 19.

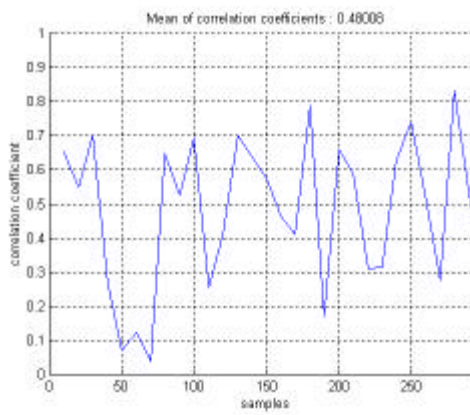
2.

.
가 가 ,
,
. (2- 19)

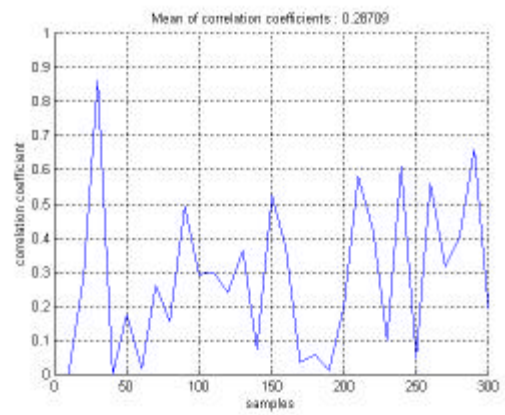
$$\rho = \frac{\langle R_1^2 \cdot R_2^2 \rangle - \langle R_1^2 \rangle \langle R_2^2 \rangle}{[(\langle (R_1^2)^2 \rangle - \langle R_1^2 \rangle^2)(\langle (R_2^2)^2 \rangle - \langle R_2^2 \rangle^2)]^{1/2}} \quad (2- 19)$$

10 (2- 19) , 1GHz
 2GHz λ 0.3 가
 $\lambda/30$ 2GHz 가
 . 가 λ

가 2- 20 2- 21

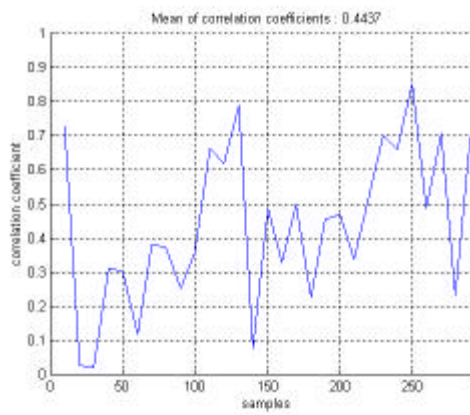


(a) $d = \lambda/30$

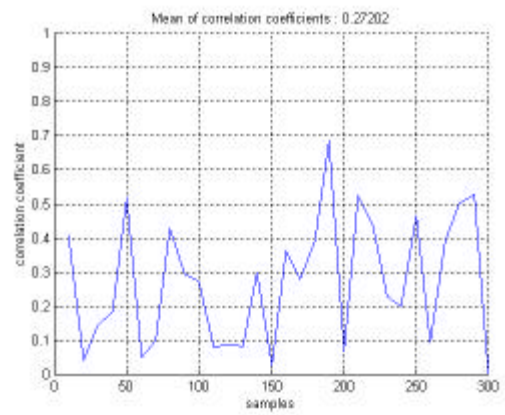


(b) $d = \lambda$

2- 20. 1GHz



(a) $d = \lambda/30$



(b) $d = \lambda$

2- 21. 2GHz

가 0.7

가

90 °

가

가

90 °

2- 22

2- 22(a) ± 45 °

가 90 °

2- 22(b)

4 × 4 MSA ± 45 °

가 90 °

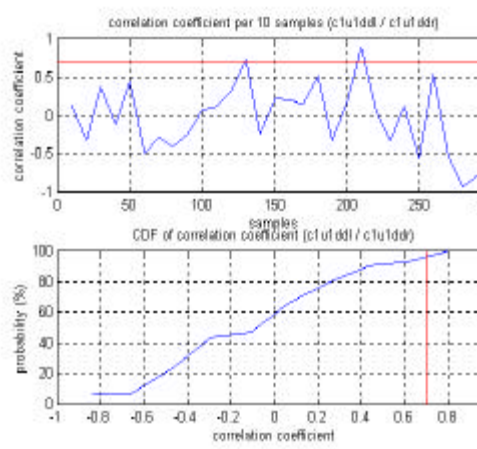
10

2- 22

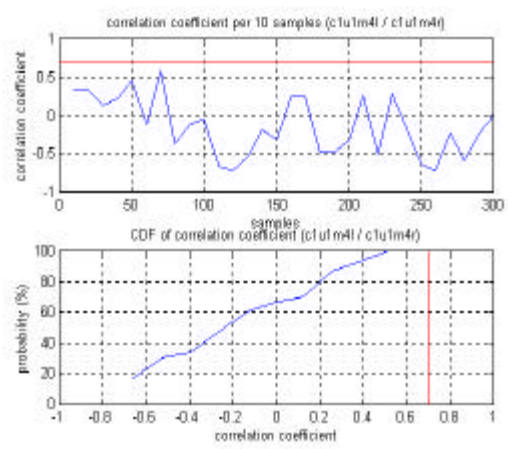
0.7

가

가



(a)



(b) 4 × 4 MSA

2- 22.

3.

가 2

가

SNR

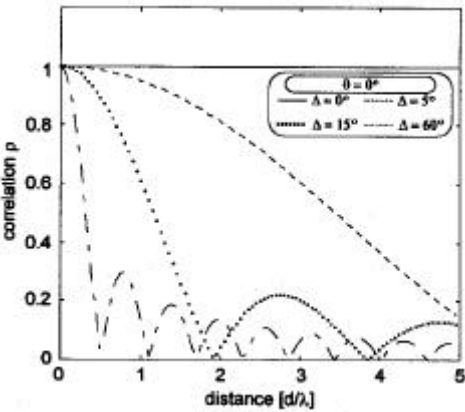
가

(DOA, direction of arrival)

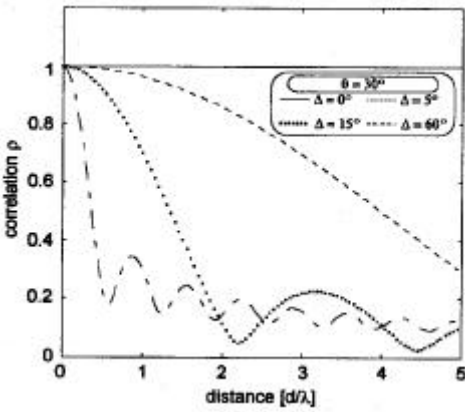
(AOS, angle of spread)

2- 23(a)

2- 23(b)

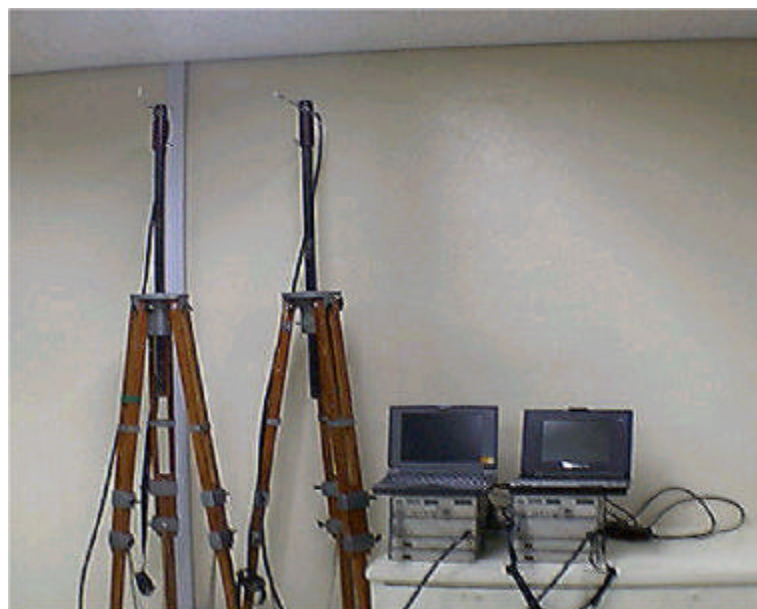
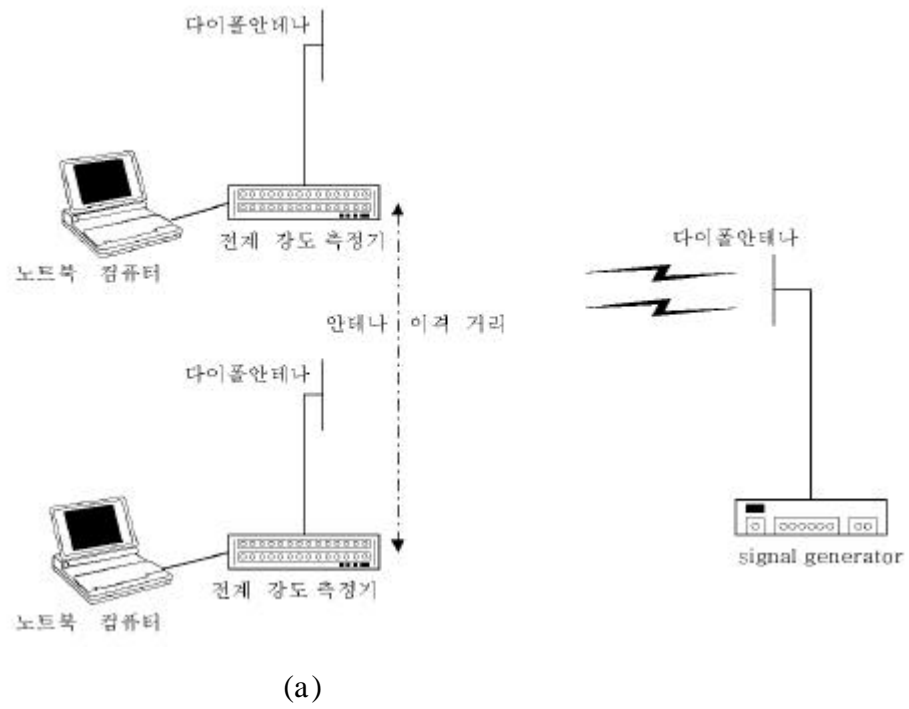


(a) DOA = 0



(b) DOA = 30°

가
2- 24



2- 24.

, 2- 24(a)

,

. 2- 24(b)

.

HP 8664A

, Anritsu

ML524B

.

GPIB

.

가 0dBm, 가 19dBm .

가 가

3

, 가

1 (6.3) .

가

,

. 가

, 10MHz .

(multi- carrier)

(frequency- hopping)

.

(rake receiver)

가

.

,

.

가 , CDMA

ISI(inter symbol interference)가 ,

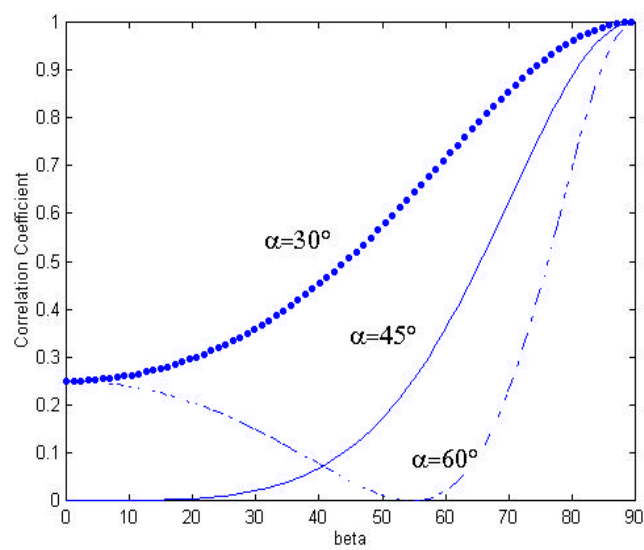
가 .

,

,

가 .

가 , XPD가 가 .
XPD가
가 .
2-25 XPD가 0 \pm



2- 25.

XPD
 ,
 가 . 가
 45° , , 가
 가 가 .
 2- 26(a)

.
 1×4, 4×4
 0° /90° ±
 45° .

(diversity gain)
 (diversity advantage) .

. 1
 가 F_l 가 F_x , P_o
 (outage probability) (2- 20) ^[5] .

$$G(P_o) = 20 \log_{10} [F_X^{-1}(P_o) / F_1^{-1}(P_o)] \tag{2- 20}$$

$G(P_o)$ P_o
 dB
 , A_o (dB)가 , (2- 21)

$$V(A_o) = F_1(10^{A_o/20}) / F_X(10^{A_o/20}) \tag{2- 21}$$

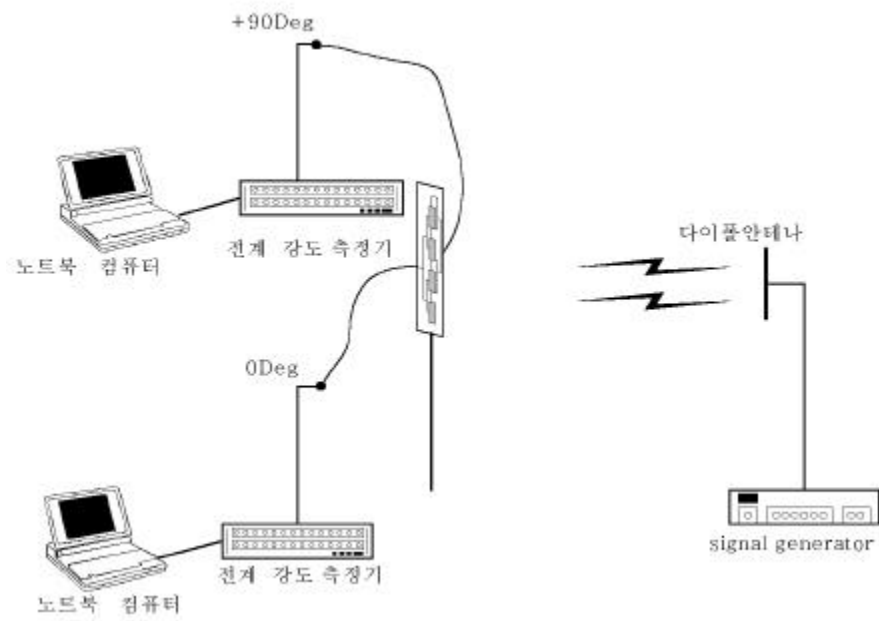
$V(A_o)$

A_o

2- 27

90% (outage probability 0.1)

$G(P_o=0.1)$



(a)

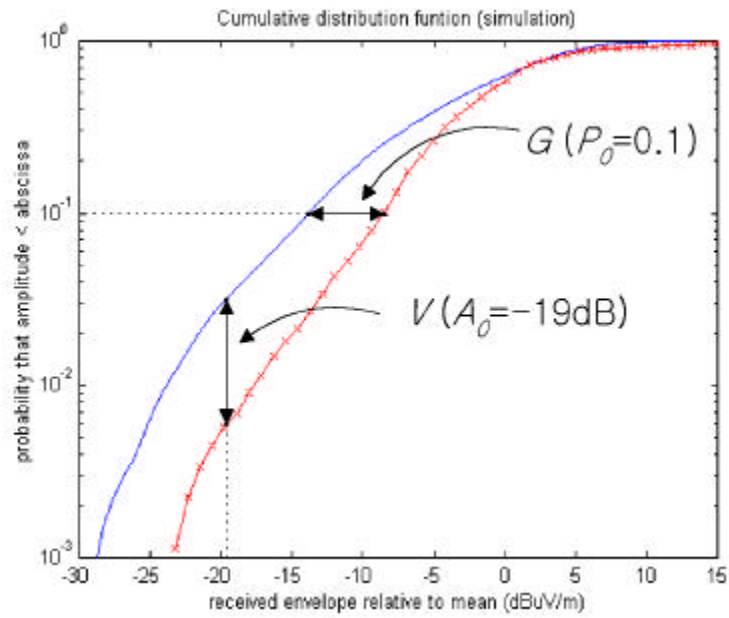


(b)

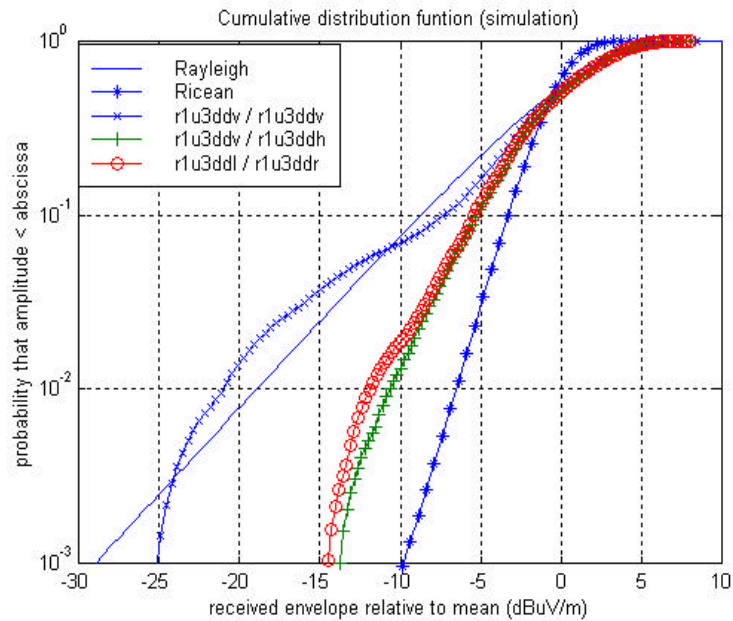
- 19dB

$V(A_0 = -19\text{dB})$

2- 28



2- 27.



(a)

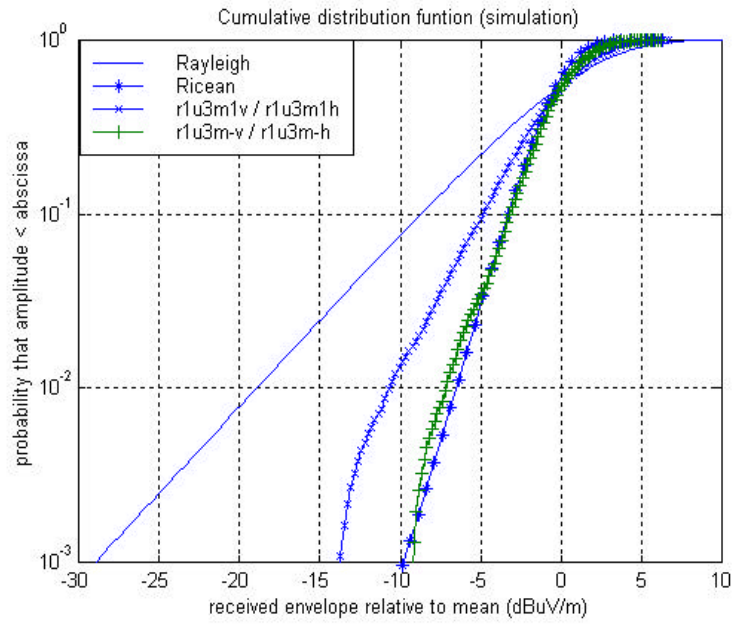
(ddv/ddv), $\theta = 90^\circ$

(ddv/ddh), $\pm 45^\circ$

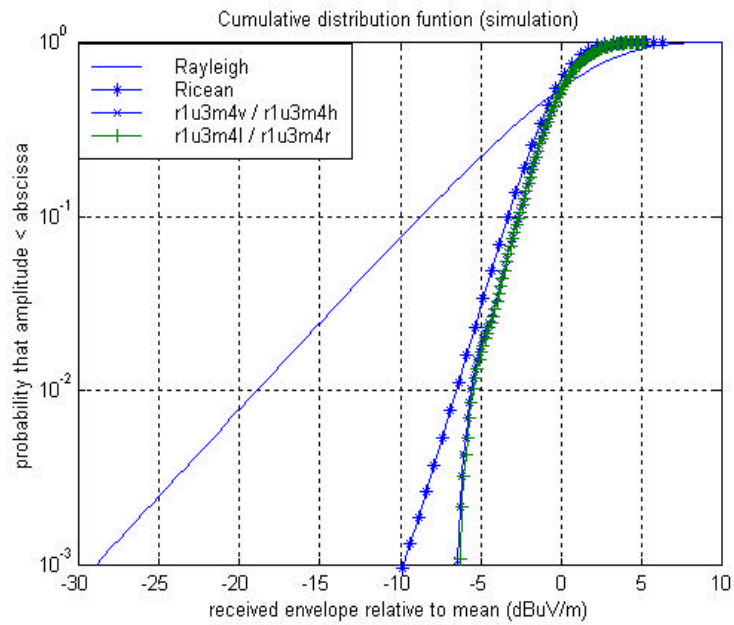
(ddl/ddr)

2- 28.

()



(b) 1×4 v- array 0° 90° MSA(m1v/m1h),
 1×4 h- array 0° 90° MSA(m_v/m_h)



(c) 4×4 0° 90° MSA(m4v/m4h),
 4×4 $\pm 45^\circ$ MSA(m4l/m4r)

(r_{lu2ddv}/r_{lu2ddv}) ,

.

Rayleigh Rician

.

2- 28 10%

(r_{lu2ddv}/r_{lu2ddv}) 4 × 4 MSA ± 45 °

(r_{lu2m4l}/r_{lu2m4r}) 6dB 가 ± 45 ° 4 × 4

MSA 6dB 가

. 1% 15dB 가 .

2- 28 가

.

3 , 1890MHz

,

(Anritsu MP663A)

,

.

4.

가

.

가 , 가

가 .

,

.

.

RF ,

.

RF

가

가

가

가

가

PN

가

PN

가

가

2- 29(a)

가

가

가

가

가

RF

ns

ps

가

가

가

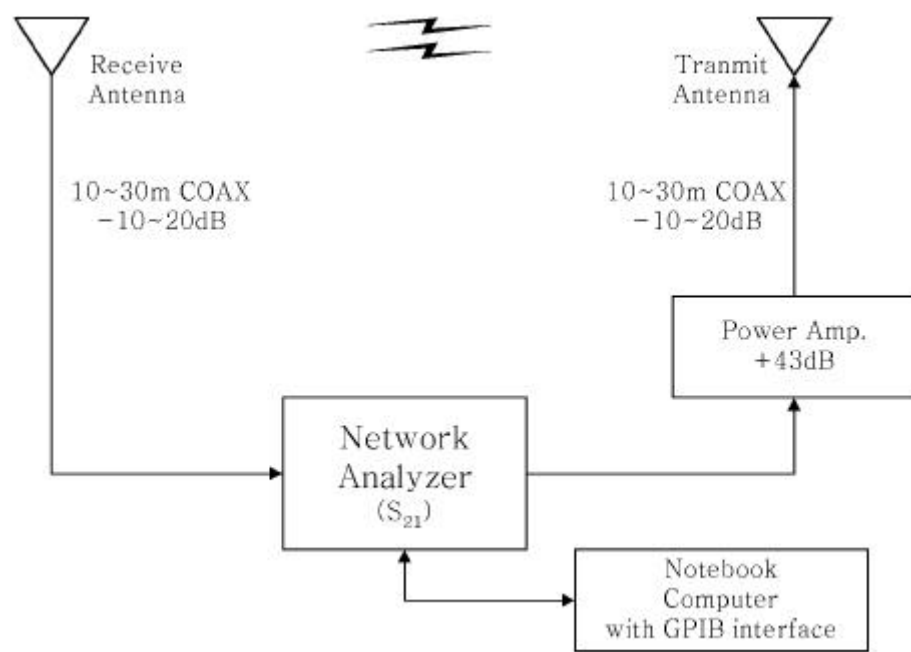
2- 29(b)

. HP8722D

B.K A2000- 20- R

Anritsu

MP663A



(a)

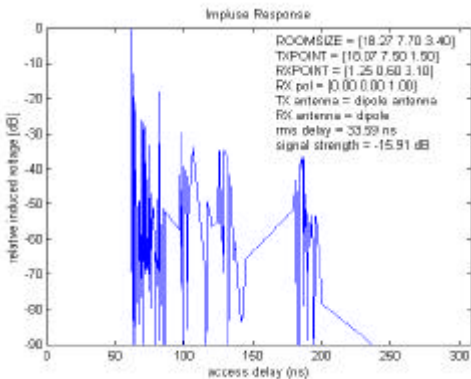


(b)

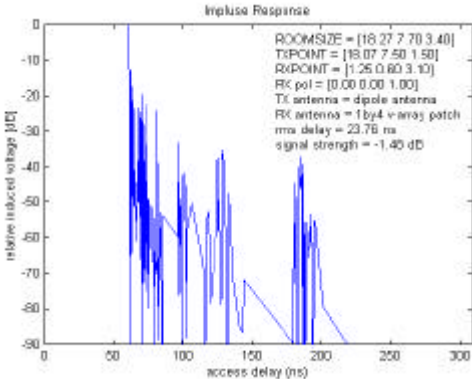
DOA(Degree Of Arrival), AOS(Angle Of Spread) . 2-3 18.27m × 7.7m
 × 3.4m RMS . RMS
 (2- 22) 2 (2nd
 central moment) .

$$\Delta = \left[\frac{\sum_{k=1}^n \tau_k^2 A_k}{\sum_{k=1}^n A_k} - \left(\frac{\sum_{k=1}^n \tau_k A_k}{\sum_{k=1}^n A_k} \right)^2 \right]^{1/2} \tag{2- 22}$$

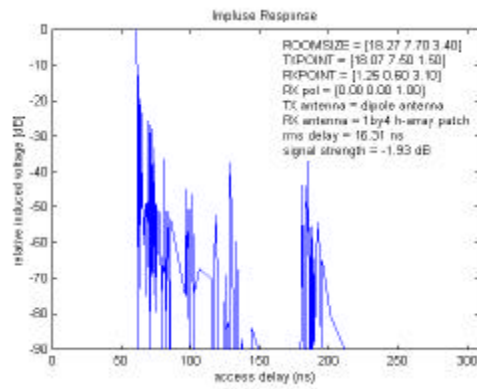
2- 30 1 × 4 MSA 4 × 4 MSA
 . 4 × 4 MSA
 . 4 × 4 MSA
 RMS 가 .
 2- 31 DOA AOS
 =40 °
 =0 ° =150 ° 150 ° AOS 가
 DOA AOS가



(a)

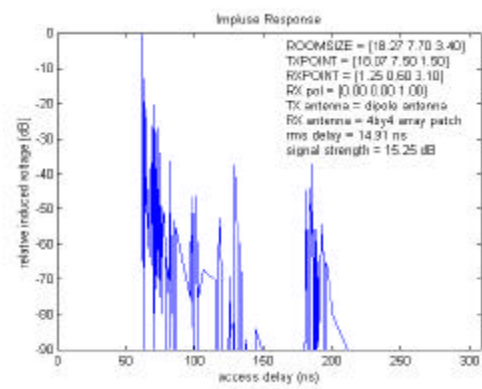


(b) 1 × 4 MSA



(c) 1×4

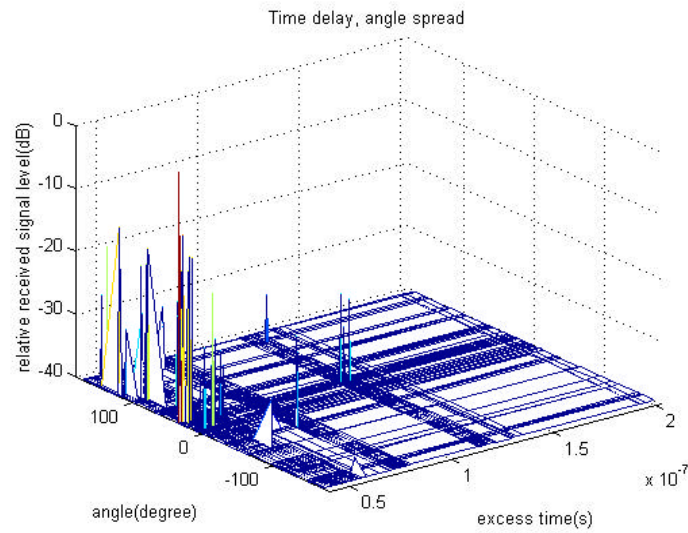
MSA



(b) 4×4

MSA

2-30.



2-31.

DOA AOS

RCS(Radar Cross Section)

[8]. ,

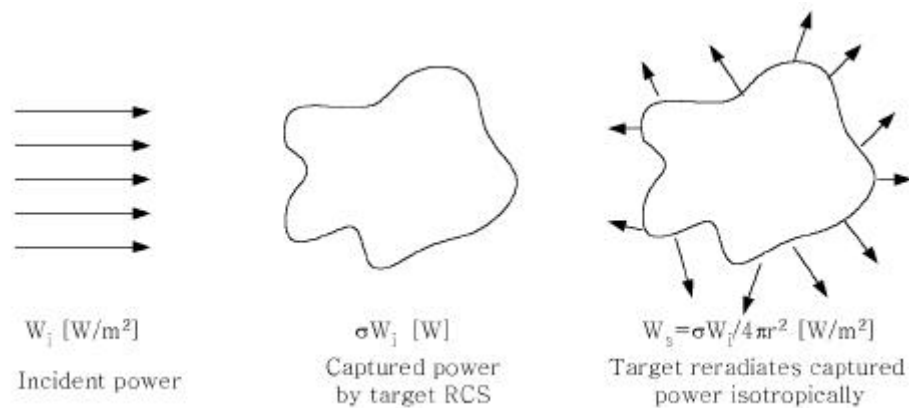
가 .

가 ,

가 . RCS

$$\sigma = \lim_{R \rightarrow \infty} \left[4\pi R^2 \frac{W_s}{W_i} \right] = \lim_{R \rightarrow \infty} \left[4\pi R^2 \frac{|E_s|^2}{|E_i|^2} \right] = \lim_{R \rightarrow \infty} \left[4\pi R^2 \frac{|H_s|^2}{|H_i|^2} \right]$$

$$\sigma = \lim_{R \rightarrow \infty} \left[4\pi R^2 \frac{W_s}{W_i} \right] = \lim_{R \rightarrow \infty} \left[4\pi R^2 \frac{|E_s|^2}{|E_i|^2} \right] = \lim_{R \rightarrow \infty} \left[4\pi R^2 \frac{|H_s|^2}{|H_i|^2} \right] \quad (3-1)$$



3- 1. RCS

$$P_r = \frac{P_t G_t}{L_t} \frac{1}{4\pi r_t^2} \sigma \frac{1}{4\pi r_r^2} \frac{G_r \lambda_0^2}{4\pi L_r} \frac{1}{L_p} \quad (3-2) \quad [8]$$

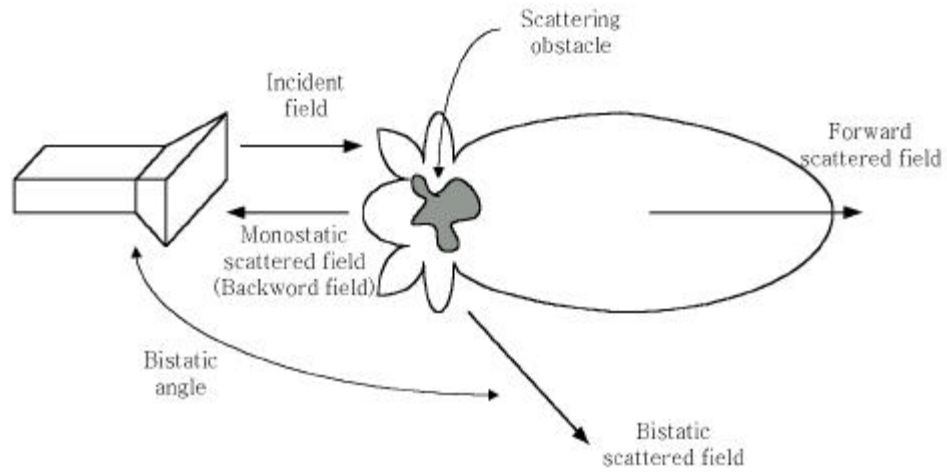
RCS
 L_t L_r , r_t r
 L_p

$$P_r = \frac{P_t G_t}{L_t} \frac{1}{4\pi r_t^2 L_{mt}} \sigma \frac{1}{4\pi r^2 L_{mr}} \frac{G_r \lambda_0^2}{4\pi L_r} \frac{1}{L_p} \quad (3-2)$$

RCS 가 . Monostatic backward RCS, forward
 RCS bistatic RCS [9],
 . Monostatic RCS
 ,
 가 . Forward RCS 180 °
 , bistatic RCS .

3-2 RCS .

가 bistatic RCS .



3-2. RCS

RCS

가 가

가

Rayleigh

RCS가 가 . resonance

RCS

가 optic ,

. Rayleigh resonance RCS

RCS

optic , (Physical

optics)

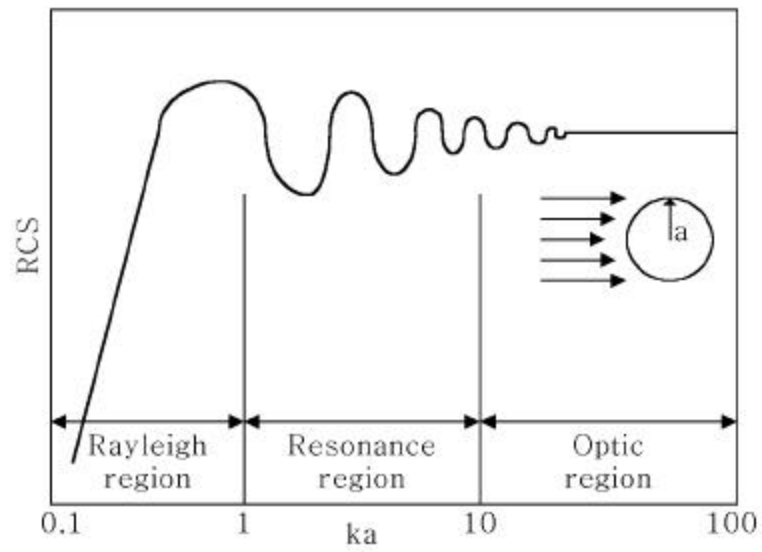
RCS

가 RCS optic

RCS

[8]

3-3 a



3-3. a RCS(k:)

2

1.

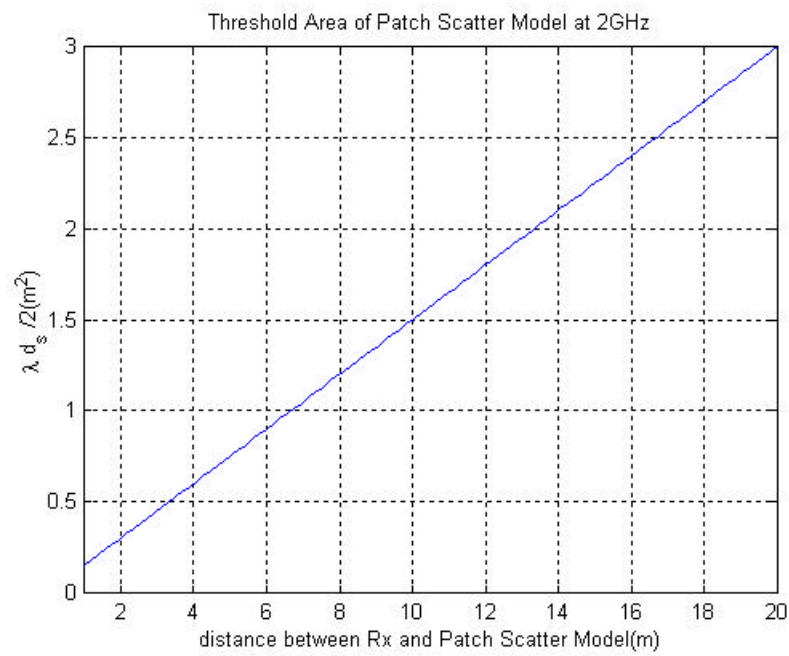
가

. (patch) 가 $(L_x L_y$
 $d_s/2)$. d_s

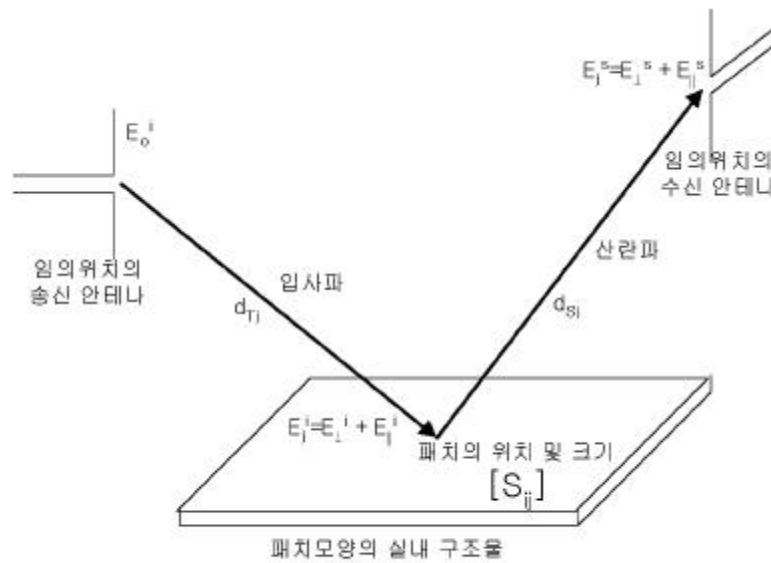
3-4

, 3-5 .
 가

가 .



3-4.



$$E^i \qquad \qquad \qquad , \quad E^s \qquad \qquad \qquad . \qquad \qquad \qquad (3-3) \qquad \qquad \qquad [9] .$$

$$\left[\begin{matrix} E^s_{\perp} \\ E^s_{\parallel} \end{matrix} \right]_j = \left[\begin{matrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{matrix} \right]_j \left[\begin{matrix} E^i_{\perp} \\ E^i_{\parallel} \end{matrix} \right]_j \qquad \qquad \qquad (3-3)$$

$$\begin{array}{l} E_{\perp} \qquad E_{\parallel} \\ \cdot j \qquad \qquad \qquad (3-4) \qquad \qquad \qquad , \\ G_{ij} \qquad \qquad \qquad j \\ \qquad \qquad \qquad , \, d_{Tj} \qquad \qquad \qquad j \\ \cdot \end{array}$$

$$\overrightarrow{E_j^i} = \overrightarrow{E_0^i} G_{ij} \exp (- \, jkd_{Tj})/4\pi d_{Tj} \qquad \qquad \qquad (3-4)$$

$$\text{bistatic RCS} \qquad \qquad \qquad .$$

$$S_{lk} = \frac{\sqrt{\sigma_{lk}}}{\sqrt{4\pi d_{sj}}} \qquad \qquad \qquad (3-5)$$

$$\text{bistatic RCS} \qquad \qquad \qquad \text{Stratton-Chu}$$

$$\begin{array}{l} [8] . \qquad \qquad \qquad \text{(far-field) 가} \\ (3-6) \qquad \qquad (3-7) \qquad \qquad \qquad . \end{array}$$

$$\overline{E_s} = ik\phi_0 \int_s \hat{s} \times [\hat{n} \times \overline{E} - Z_0 \hat{s} \times (\hat{n} \times \overline{H})] e^{ik\overline{r} \cdot (\hat{i}^\wedge \hat{s}^\wedge)} dS \qquad \qquad \qquad (3-6)$$

$$\overline{H_s} = ik\phi_0 \int_s \hat{s} \times [\hat{n} \times \overline{H} - Y_0 \hat{s} \times (\hat{n} \times \overline{E})] e^{ik\overline{r} \cdot (\hat{i}^\wedge \hat{s}^\wedge)} dS \qquad \qquad \qquad (3-7)$$

$$\hat{n} \qquad \qquad \qquad , \quad \hat{i} \qquad \hat{s} \qquad \qquad \qquad , \quad Z_0$$

$$Y_0 \qquad \qquad \qquad \overline{r}$$

$$\phi_0 = \exp(ikR)/4\pi R$$

$$\overline{H_s} = Y_0 \hat{s} \times \overline{E_s}$$

$$\overline{E_s} = -i2kZ_0 H_0 \phi_0 \int_s \hat{s} \times [\hat{s} \times (\hat{n} \times \hat{h}_i)] e^{-ik\bar{r} \cdot (\hat{i} - \hat{s})} dS$$

$$\hat{n} \times \overline{E} = 0$$

$$\hat{n} \times \overline{H} = 2\hat{n} \times \overline{H}_i$$

$$\overline{E_s} = -i2kZ_0 H_0 \phi_0 \int_s \hat{s} \times [\hat{s} \times (\hat{n} \times \hat{h}_i)] e^{-ik\bar{r} \cdot (\hat{i} - \hat{s})} dS$$

$$\hat{h}_i$$

$$\text{RCS}$$

$$\text{RCS}$$

$$\text{RCS}$$

$$\phi_0$$

$$\exp(ikR)$$

$$\text{RCS}$$

$$\hat{e}_r$$

$$\sqrt{\sigma} = \lim_{R \rightarrow \infty} 2\sqrt{\pi R} \frac{\overline{E_s} \cdot \hat{e}_r}{E_0} e^{ikR}$$

$$\sqrt{\sigma} = -i \frac{k}{\pi} \int_s \hat{n} \cdot \hat{e}_r \times \hat{h}_i e^{ik\bar{r} \cdot (\hat{i} - \hat{s})} dS$$

$$\sigma_{lk} = \frac{k^2}{\pi} (L_x L_y) \left(\frac{\sin(k \xi_x L_x / 2)}{k \xi_x L_x / 2} \right)^2 \left(\frac{\sin(k \xi_y L_y / 2)}{k \xi_y L_y / 2} \right)^2 |\gamma_{lk}|^2 \quad (3-13)$$

$$\xi_x = \sin \theta_i \cos \varphi_i - \sin \theta_s \cos \varphi_s$$

$$\xi_y = \sin \theta_i \sin \varphi_i - \sin \theta_s \sin \varphi_s$$

$$\gamma_{11} = \sin \theta_s \sin \theta_i \cos \theta_i (\sin \varphi_s \sin \varphi_i + \cos \varphi_s \cos \varphi_i) \rho_s \Gamma_{\perp}(\theta_i) \quad (3-14)$$

$$\gamma_{12} = \sin \theta_s \cos \theta_s \sin \theta_i \cos \theta_i (\sin \varphi_s \cos \varphi_i - \cos \varphi_s \sin \varphi_i) \rho_s \Gamma_{\parallel}(\theta_i)$$

$$\gamma_{21} = \sin \theta_s \sin \theta_i (\cos \varphi_s \sin \varphi_i - \sin \varphi_s \cos \varphi_i) \rho_s \Gamma_{\perp}(\theta_i)$$

$$\gamma_{22} = \sin \theta_s \cos \theta_s \cos \theta_i (\cos \varphi_s \cos \varphi_i + \sin \varphi_s \sin \varphi_i) \rho_s \Gamma_{\parallel}(\theta_i)$$

$$(3-14) \quad \xi_x \quad \xi_y \quad x \quad y$$

$$, \quad \gamma_{lk}$$

$$\rho_s \quad (3-15)$$

$$\sigma_h \quad I_0[] \quad 0 \quad [15].$$

$$\rho_s = I_0 \left[8 \left(\frac{\pi \sigma_h \sin \theta_i}{\lambda} \right)^2 \right] \cdot \exp \left[- 8 \left(\frac{\pi \sigma_h \sin \theta_i}{\lambda} \right)^2 \right] \quad (3-15)$$

$$, \quad ,$$

$$,$$

$$,$$

$$,$$

$$3-7$$

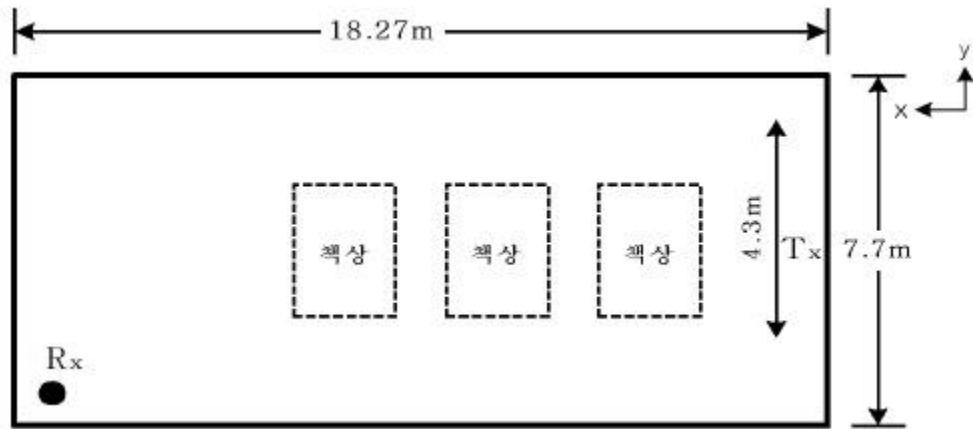
$$18.27\text{m} \times 7.7\text{m} \quad 3$$

$$. \quad y \quad 4.3\text{m} \quad 3-7$$

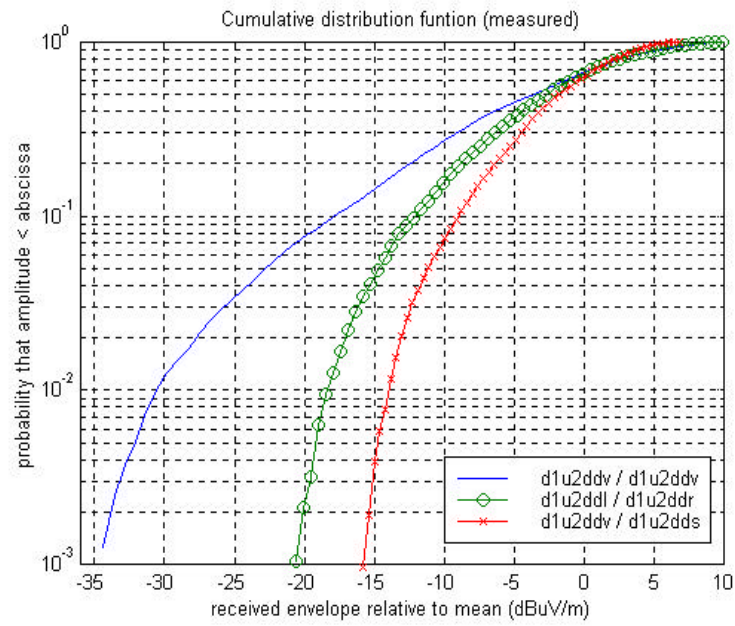
$$. \quad 3-8 \quad 3-7$$

$$3-8$$

$$가$$



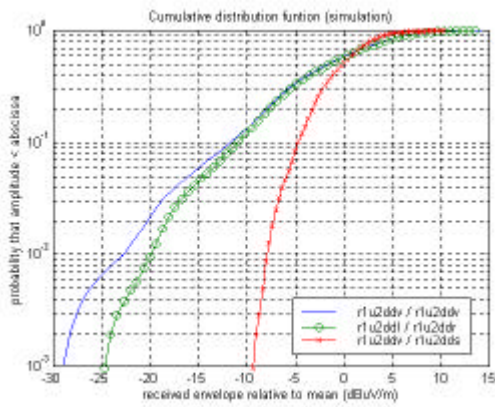
3-7.



(a)

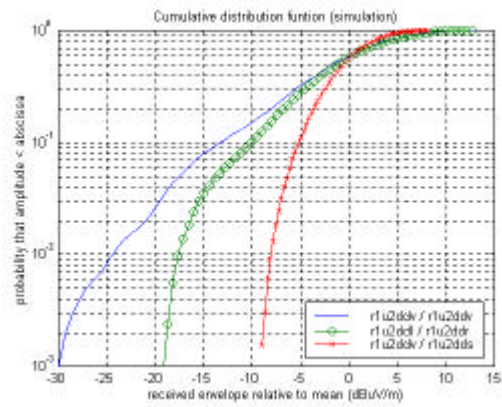
3-8.

()



(b)

()



(c)

()

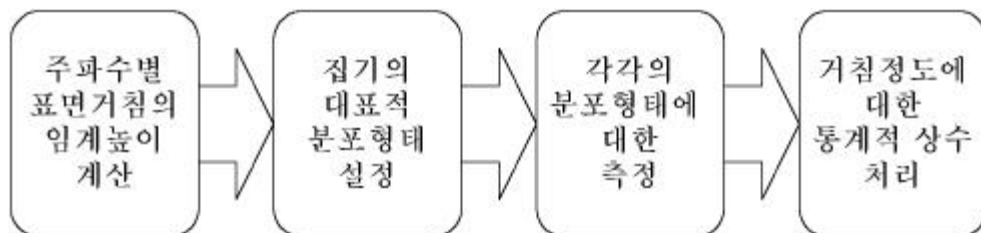
3-8.

2.

가

가

3-9



3-9.

가

가

가

$$h_c = \frac{\lambda}{8 \sin \theta_i}$$

$$\rho_s = I_0 \left[8 \left(\frac{\pi \sigma_h \sin \theta_i}{\lambda} \right)^2 \right] \cdot \exp \left[- 8 \left(\frac{\pi \sigma_h \sin \theta_i}{\lambda} \right)^2 \right] \quad (3-16)$$

$$\Gamma_{\perp rough}(\theta_i) = \rho_s \Gamma_{\perp}(\theta_i)$$

$$\Gamma_{\parallel rough}(\theta_i) = \rho_s \Gamma_{\parallel}(\theta_i)$$

(3-16)

(3-13)

3-10

1.89GHz

0°

가

가

, 90°

가

0.02m

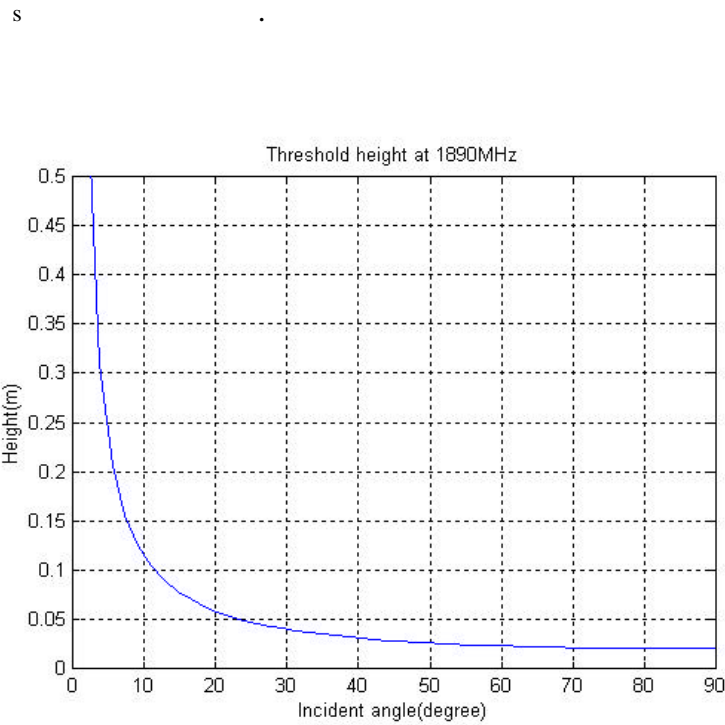
가

60°

가

2.5cm

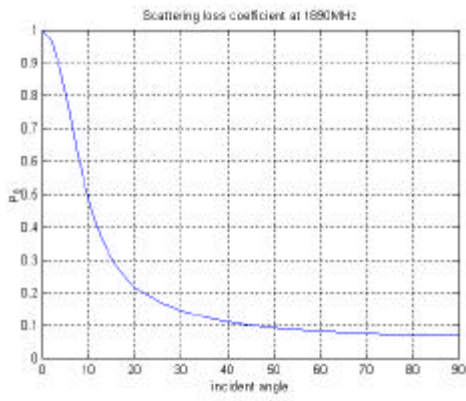
가



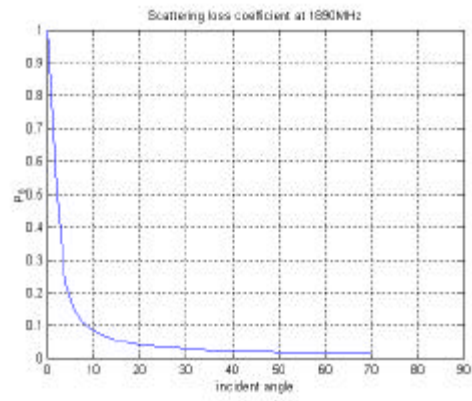
3- 10.

3- 11

가



(a) 0.1



(b) 0.5

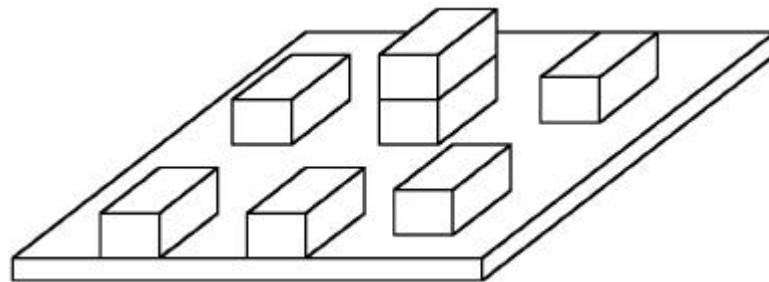
3- 11.

가

가

가

3- 12



3- 12.

가

가

가

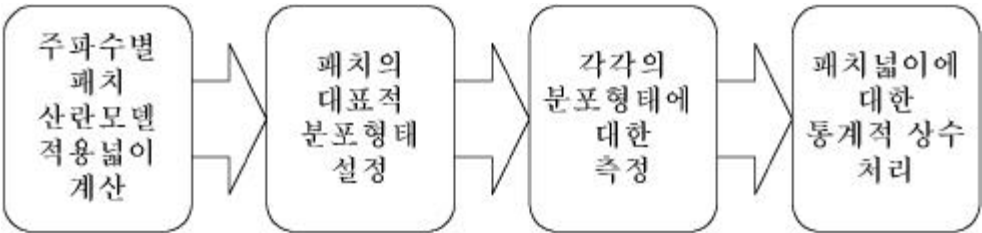
가

3.

$L_x L_y \quad d_s/2$
가 . 가

가

3- 13

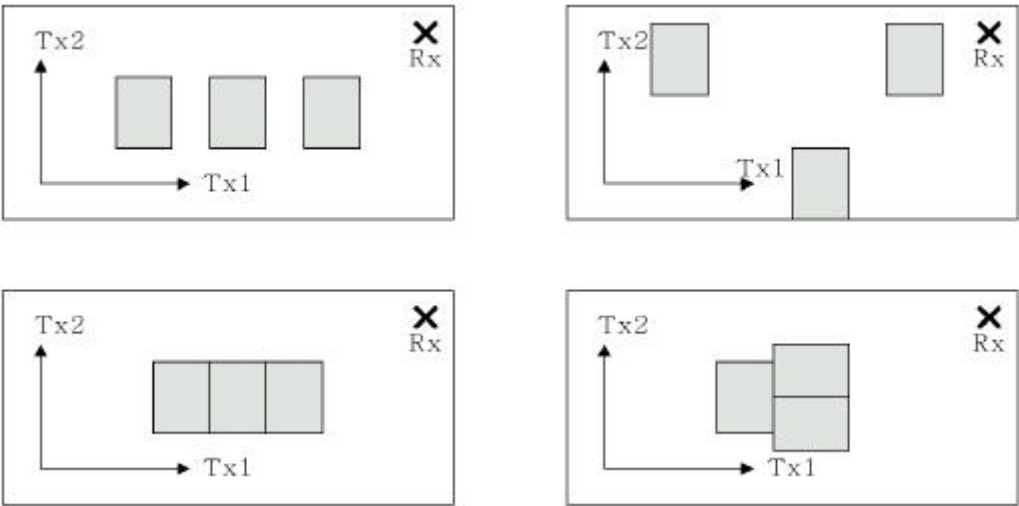


3- 13.

1890MHz

3-4

3- 14



3- 14.

3- 14

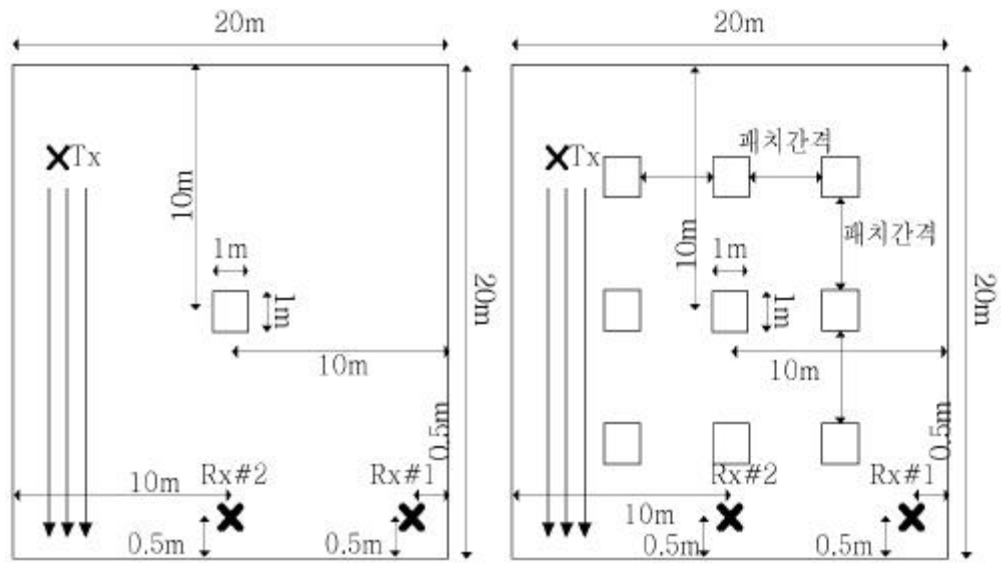
가 가 , .
4 .

4.

가
3- 1
3- 15 .
가 20m × 20m 50 × 50
3- 16 3- 15(a) (10m,10m,0.5m) 1m × 1m
.
 $\theta_i = \theta_s$ $\varphi_i = \pi + \varphi_s$ 가 .
가 .
RCS가 sinc . Rx
가 .
3- 17 3- 15(b) 3m
가 가
 , 가 .

3- 1.

	1890MHz
	20m × 20m × 3m
	1.4m
	Rx#1(0.5m,0.5m,2.5m) Rx#2(0.5m,10m,2.5m)



(a) 가 1
3- 15.

(b) 가 9

3- 18 3- 19

0.2m × 0.2m

3- 16

3- 17

가

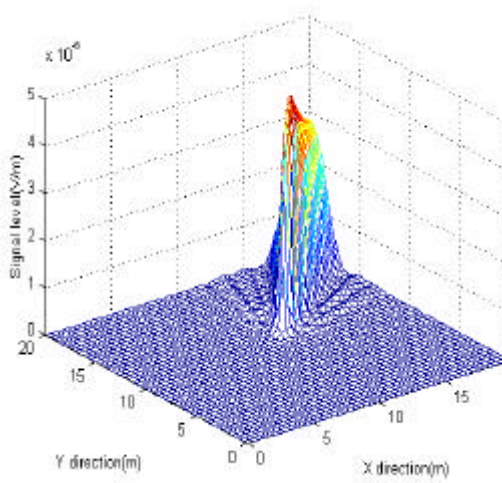
가

3- 20 3- 21

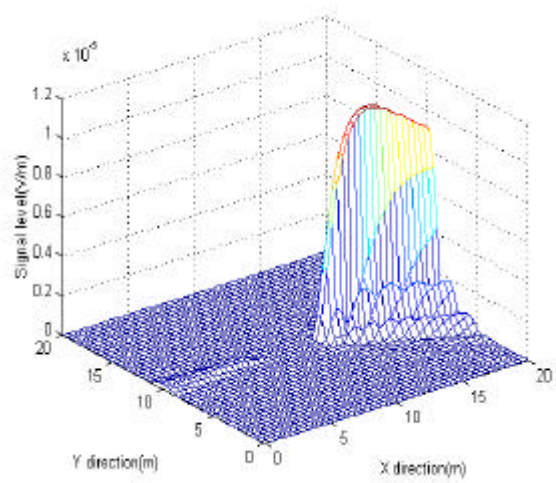
가

가 가

가



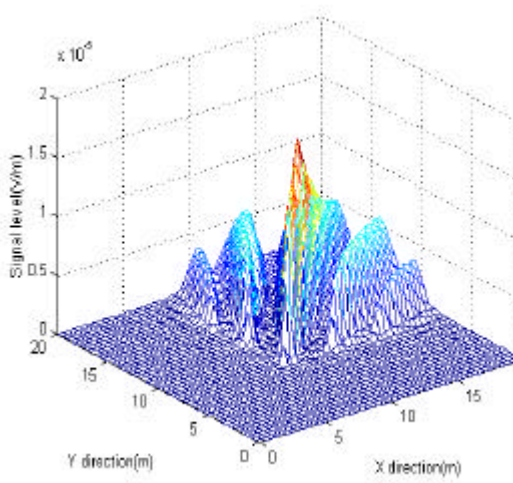
(a) Rx#1



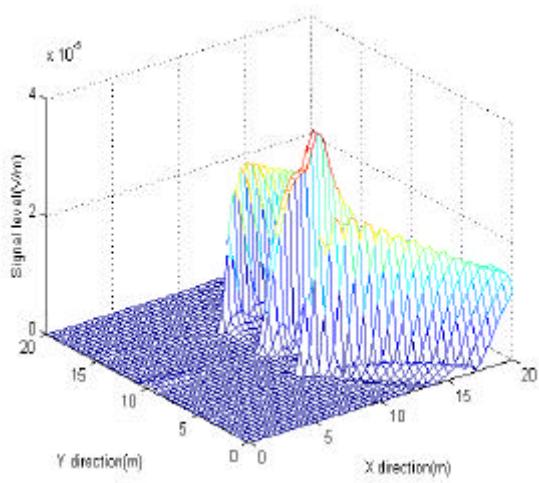
(b) Rx#2

3- 16. 1

(:1m × 1m)



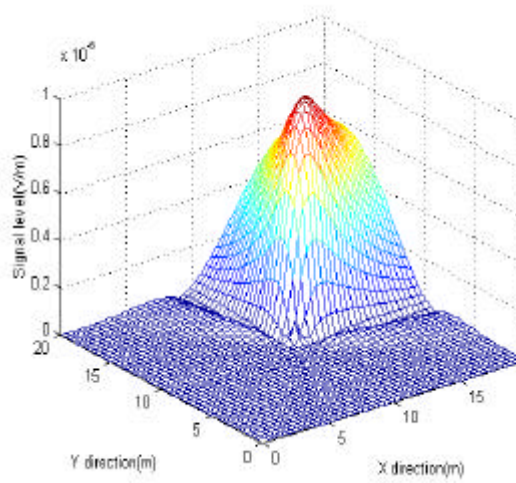
(a) Rx#1



(b) Rx#2

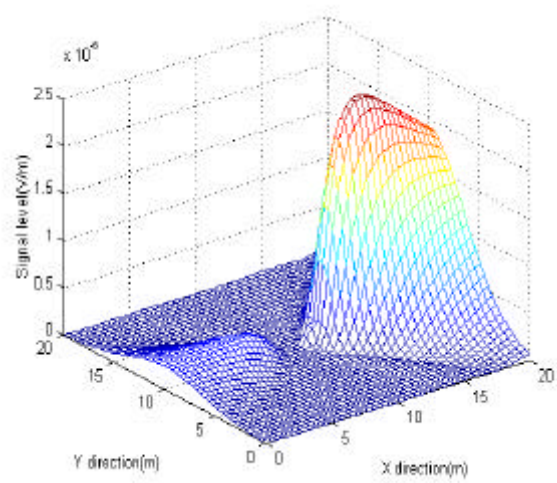
3- 17. 9

(:1m × 1m, :3m)



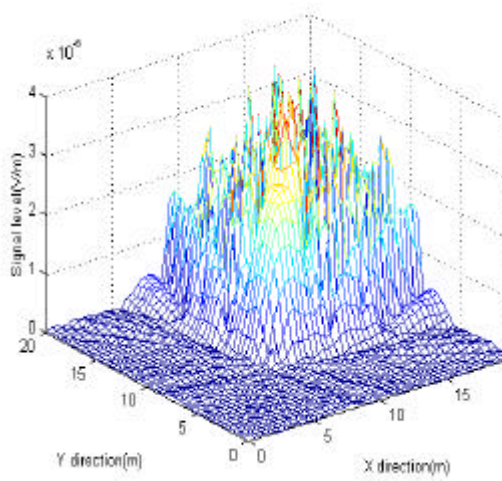
(a) Rx#1

3- 18. 1



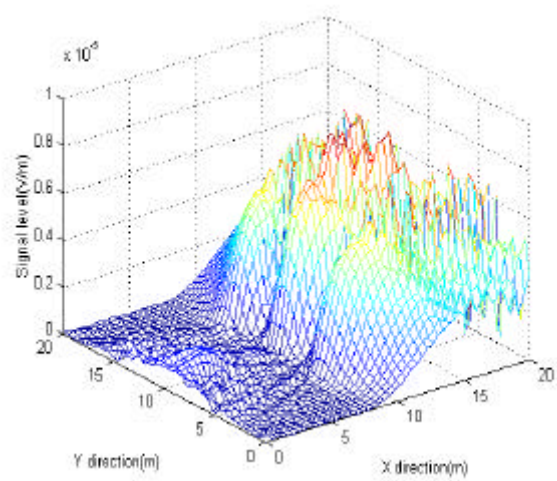
(b) Rx#2

(:0.2m × 0.2m)



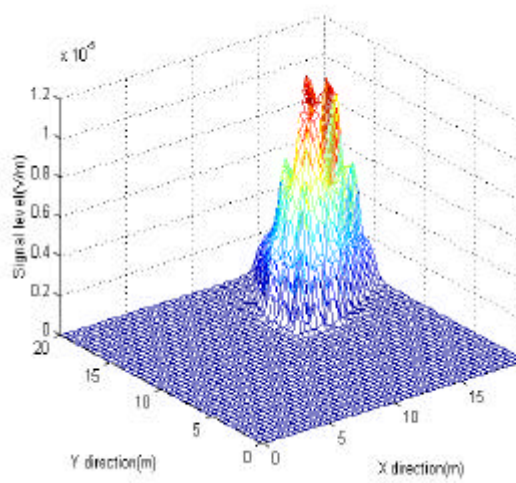
(a) Rx#1

3- 19. 9



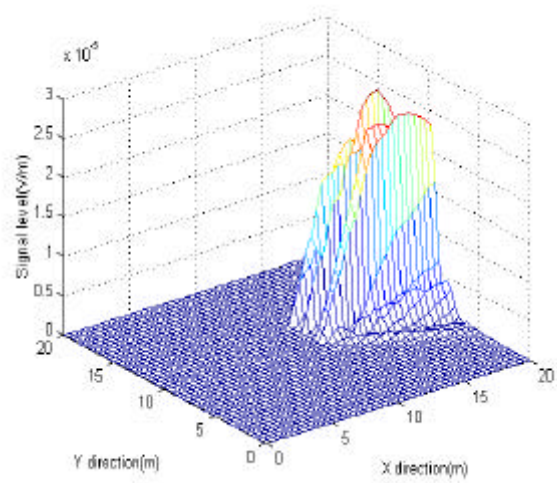
(b) Rx#2

(:0.2m × 0.2m, :3m)



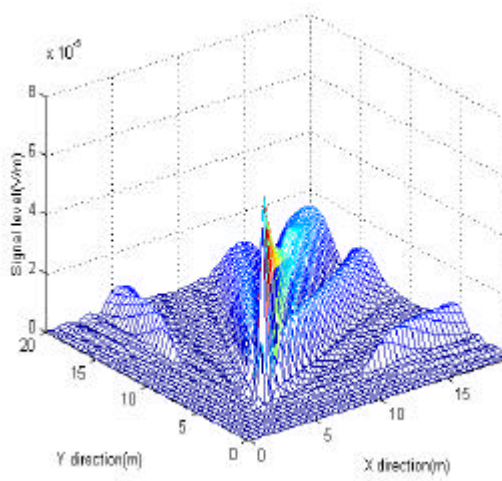
(a) Rx#1

3-20. 9



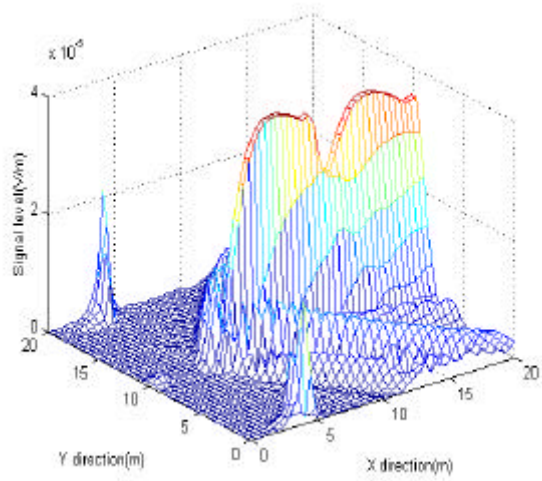
(b) Rx#2

(:1m × 1m, :1.1m)



(a) Rx#1

3-21. 9



(b) Rx#2

(:1m × 1m, :7m)

4

2

3

가

,

•

1

3

• , RCS

•

가

, RCS

[10]

RCS

20dB

•

2

• (4- 1)

$$, \rho_{ic} \quad .$$

$$E_s' = \rho_{ic} E_s \quad (4-1)$$

4-1

가

3

6

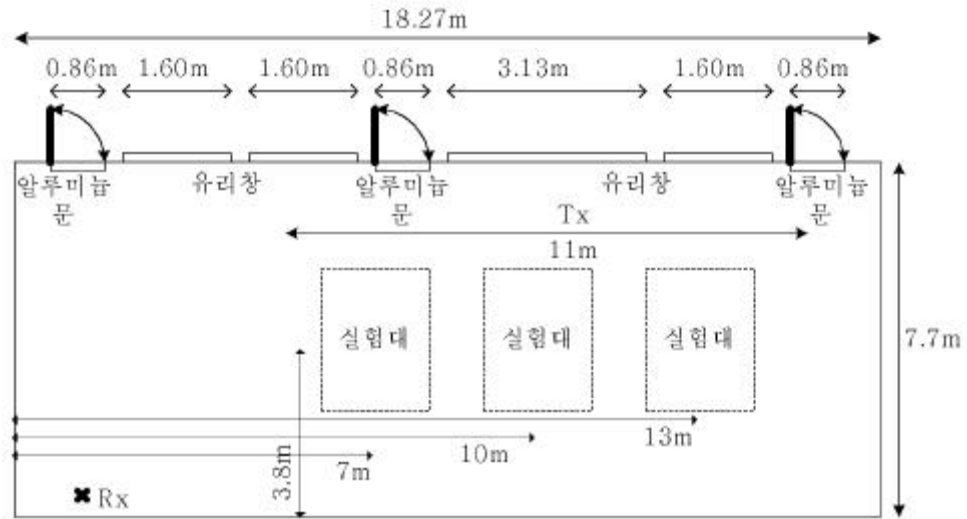
•

가

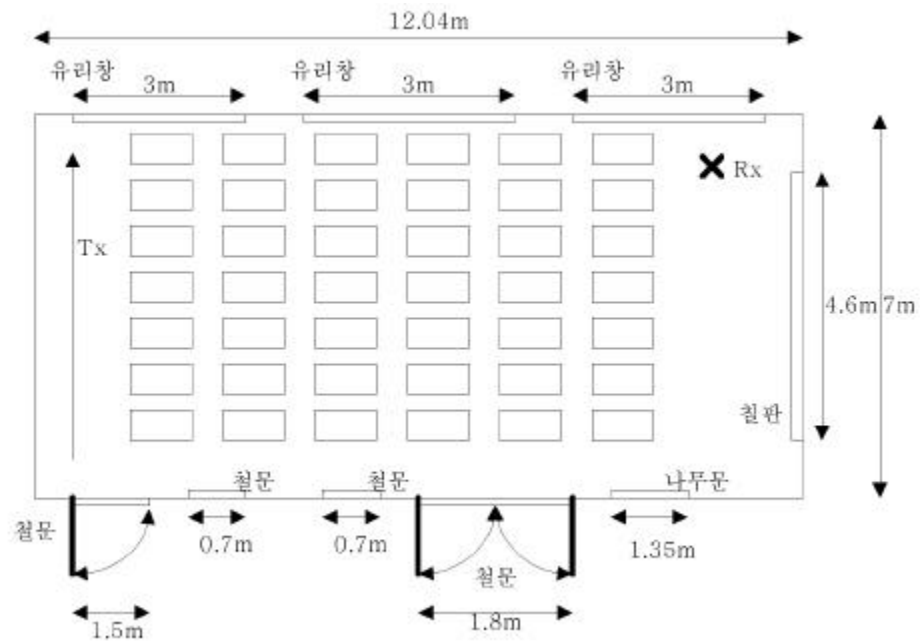
32

64 가

4- 1



(a)



(b)

4- 1.

4- 1.

	1890MHz	1890MHz
	18.27m × 7.7m × 3.4m	12.04m × 7m × 3.3m
(m)	(16.6,6,1.27) (5.6,6,1.27)	(10.84,0.56,1.67) (10.84,5.87,1.67)
(m)	(1.24, 0.6, 2.4)	(1.2, 0.7, 2.45)
	HP8664A	HP8664A
	Anritsu ML524B	Anritsu ML524B
	3 , (1/2) 6	32 , 64

4- 2 .

가 ,

.

.

가

4- 3 x y

,

3 .

가

80dB 가 . RCS

.

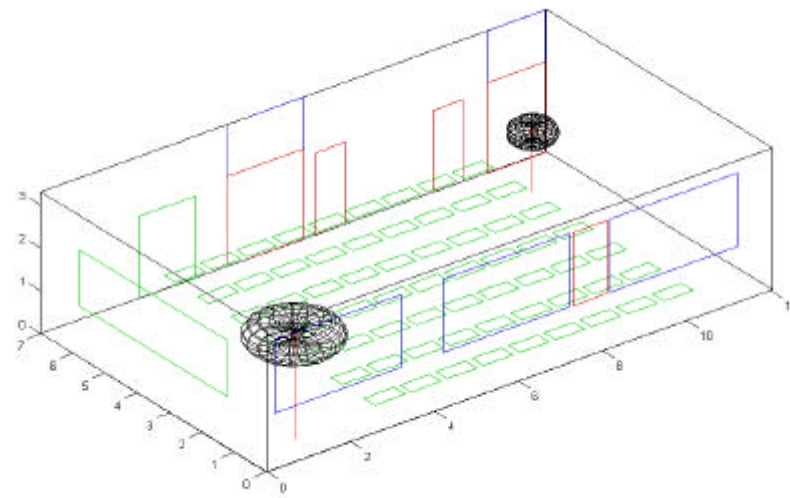
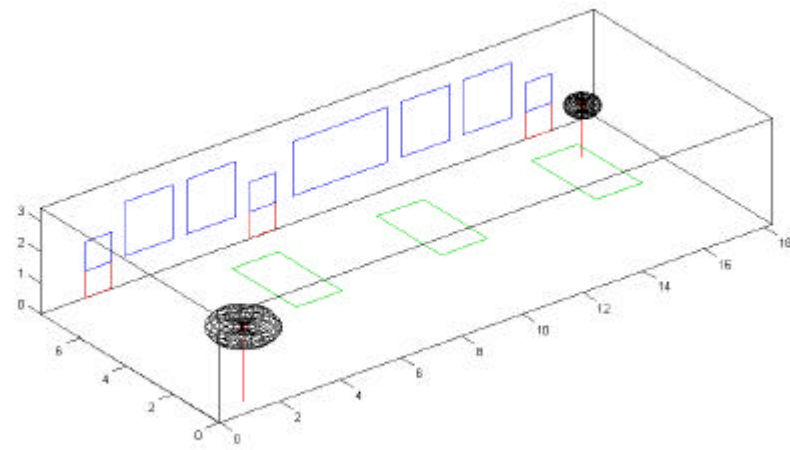
10, 100, 1000

100

가

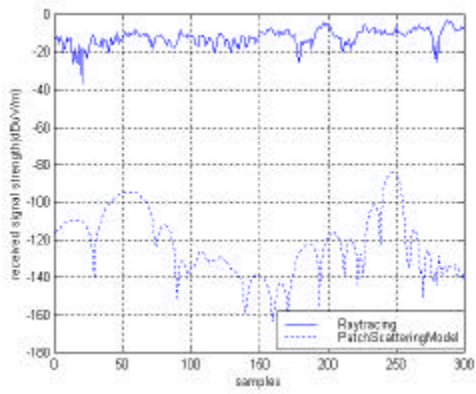
.

,

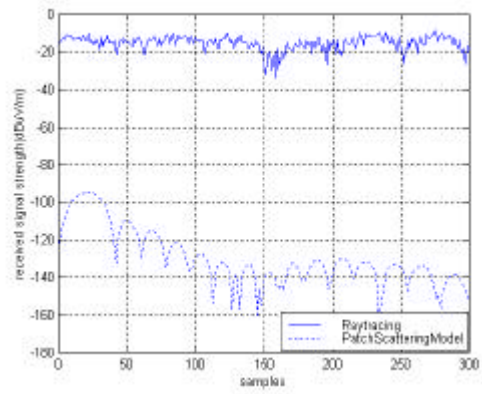


(b) (64)

4-2.



(a) x



(b) y

4-3.

가 , 가

0

100

가 가

가 가

가

4-4

, 가

1%

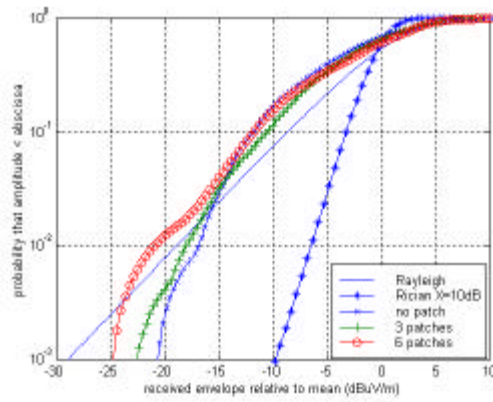
5dB

가

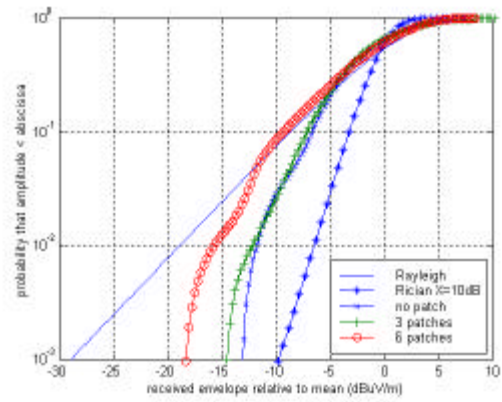
4-5

100

가

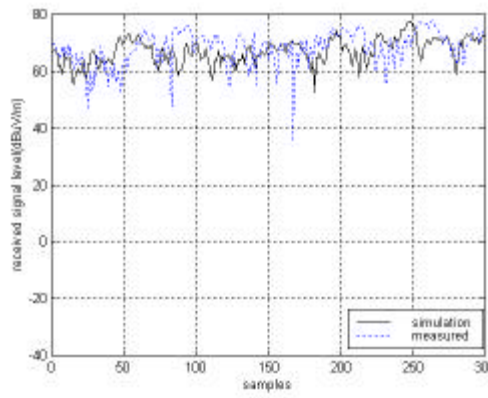


(a)

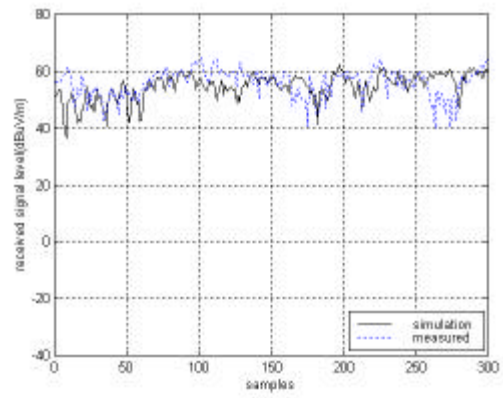


(b)

4-4.



(a) 가 3



(b) 가 6

4-5.

가
3
32 64
0
0.5m 0.7m
1/2 가 0.01

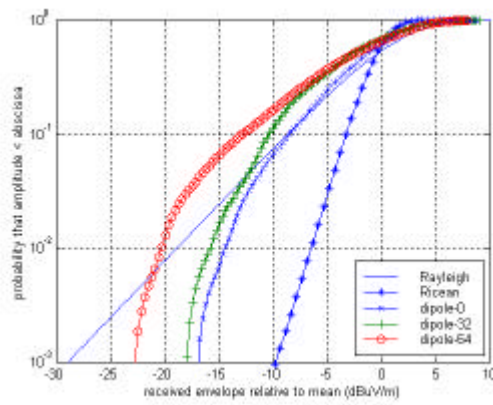
4-6

4-7

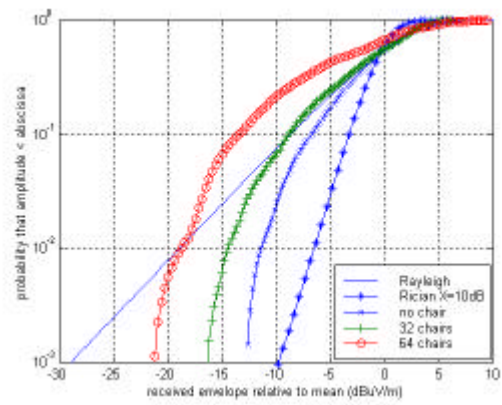
2

100

100

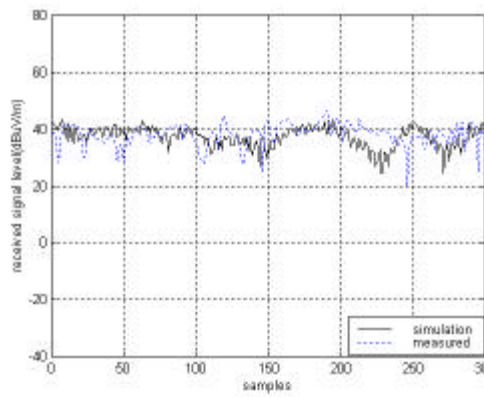


(a)

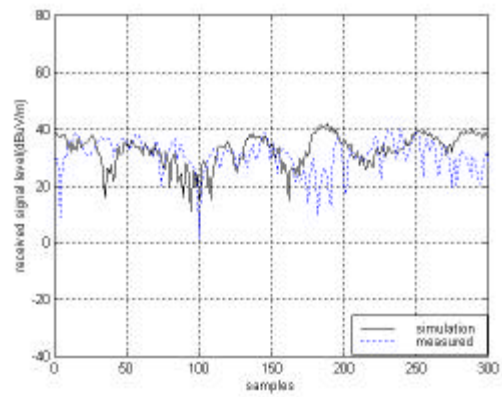


(b)

4-6.



(a) 32



(b) 64

4-7.

2

1.

4-8

1890MHz

HP

8664A

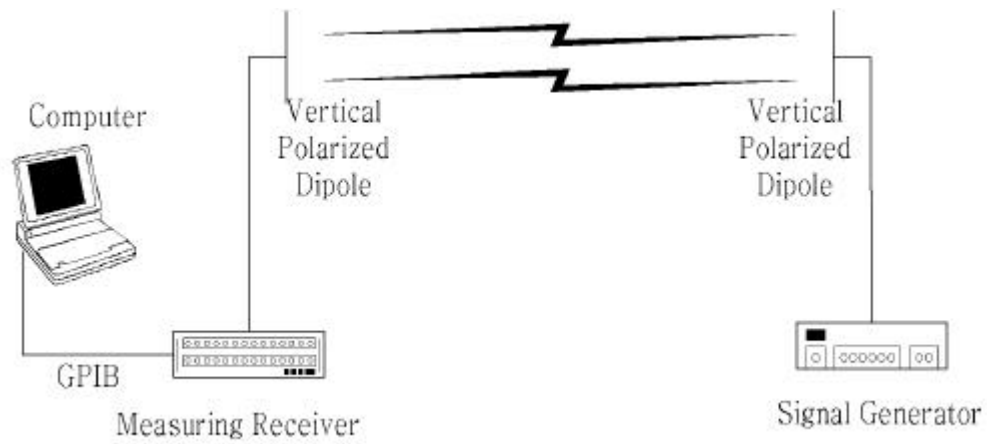
Anritsu

ML524B

GPIO

1.67m,

2.44m



4-8.

2.

4

4-2

3

1, 2, 3

가

가

,

4

가

가 . 4-9 4- 11
1, 2, 4 .

4- 2.

		(m)	
1		$21.63 \times 8 \times 2.88$	‘ㄷ’
2		$17.73 \times 8.65 \times 3.4$ 3	14 1
3		$9.95 \times 13.53 \times 2.5$ 5	,
4		$83.55 \times 1.98 \times 2.7$ 1	



4- 9.



4- 10.

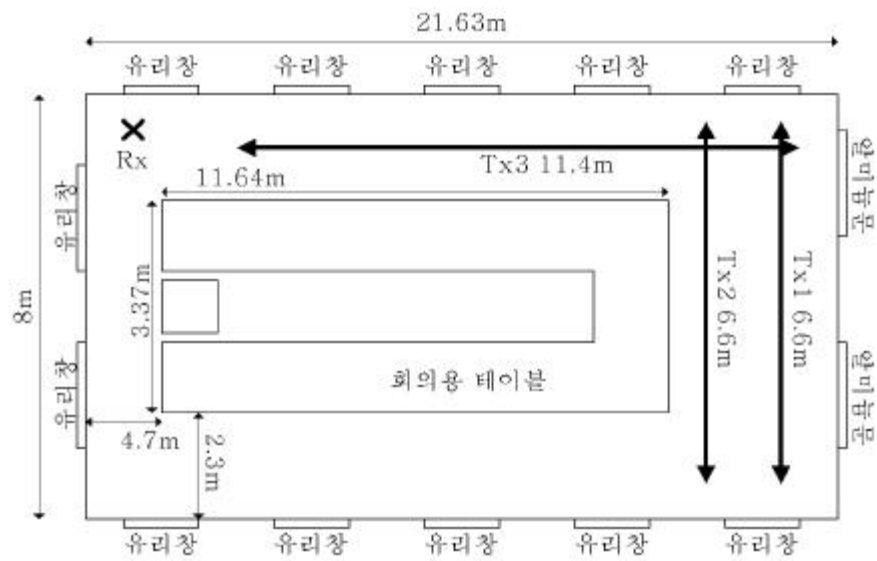


4- 11.

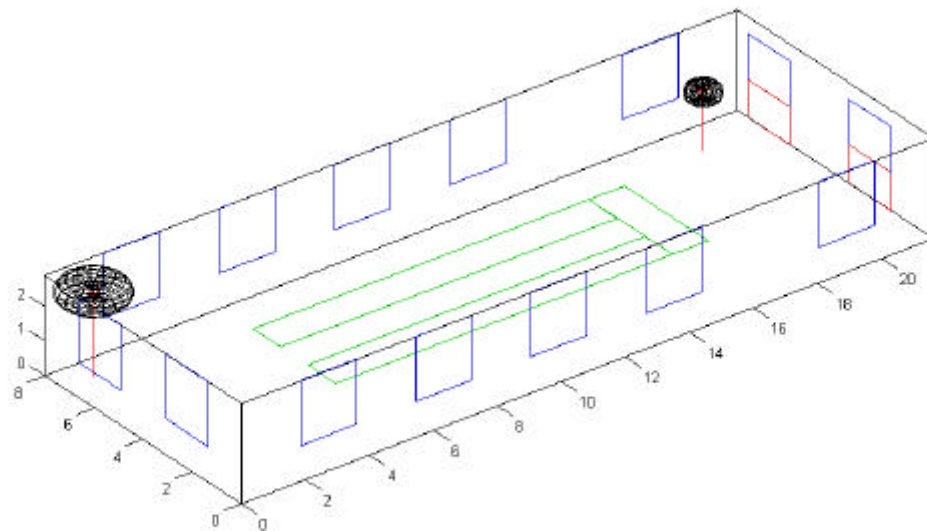
4- 12

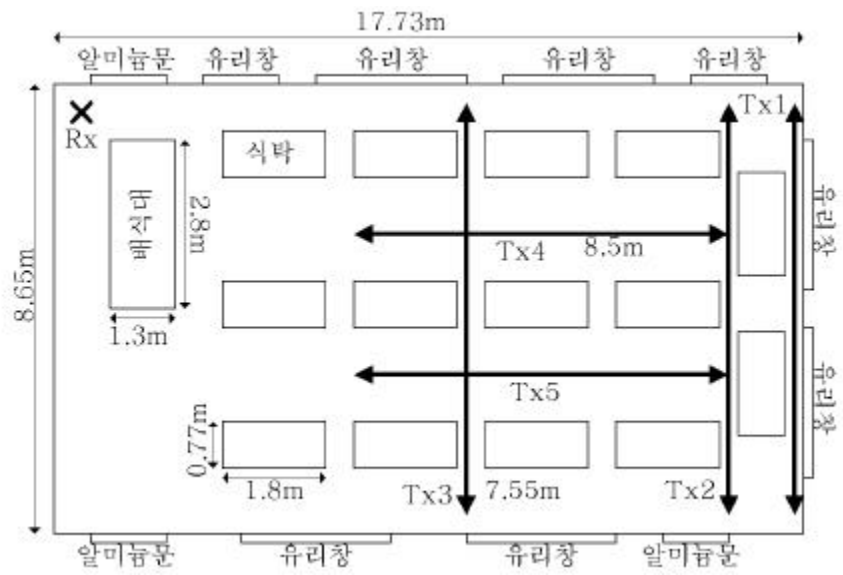
4- 15

5 , 3 1 3 , 2
 , 3 1 4 2

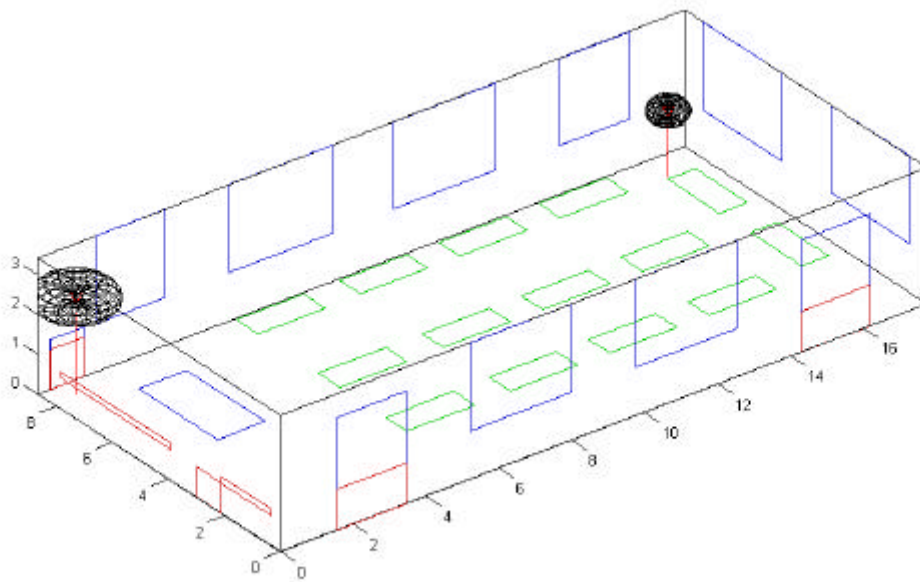


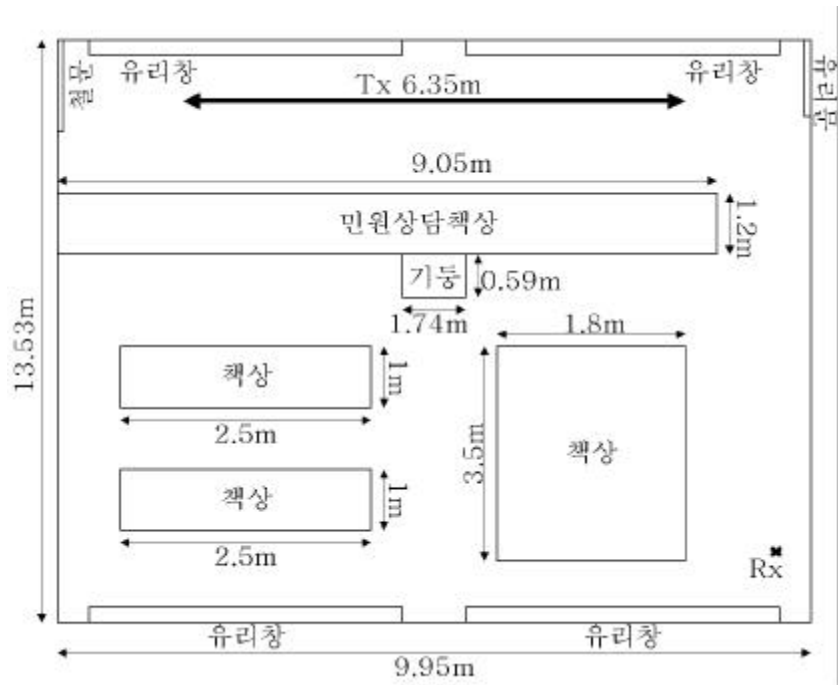
(a)



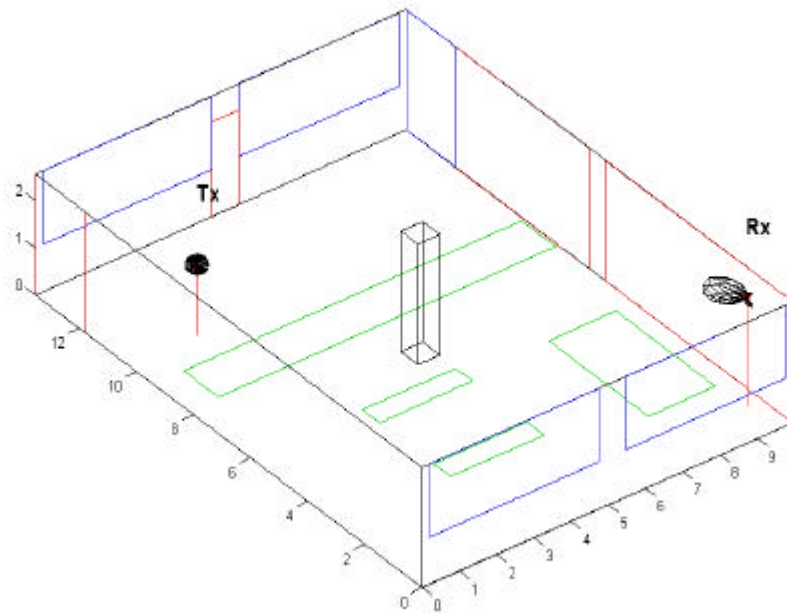


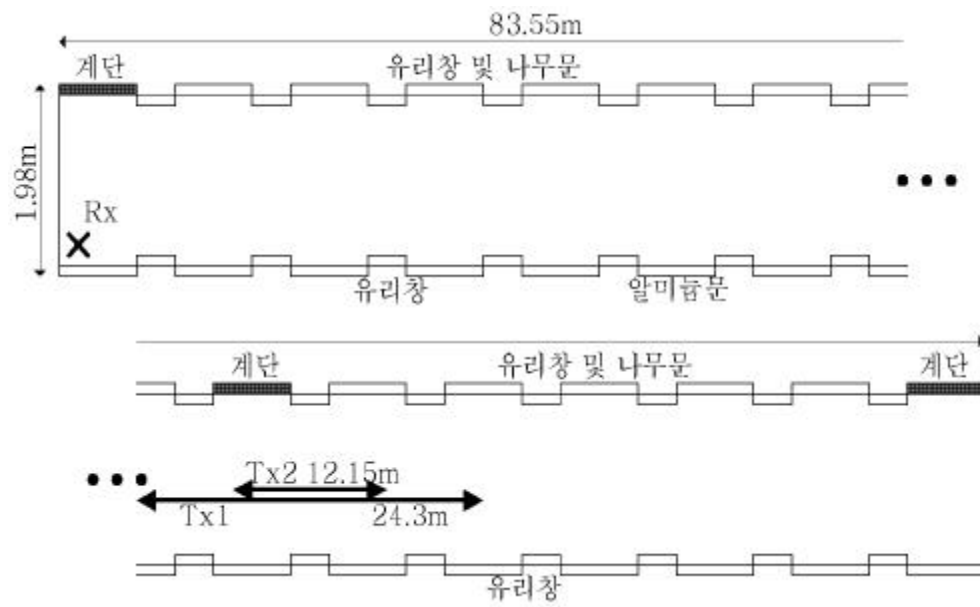
(a)



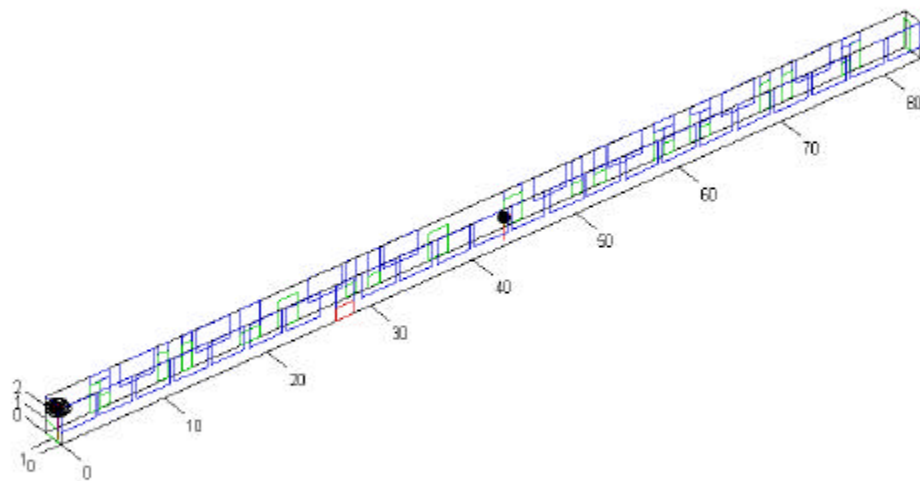


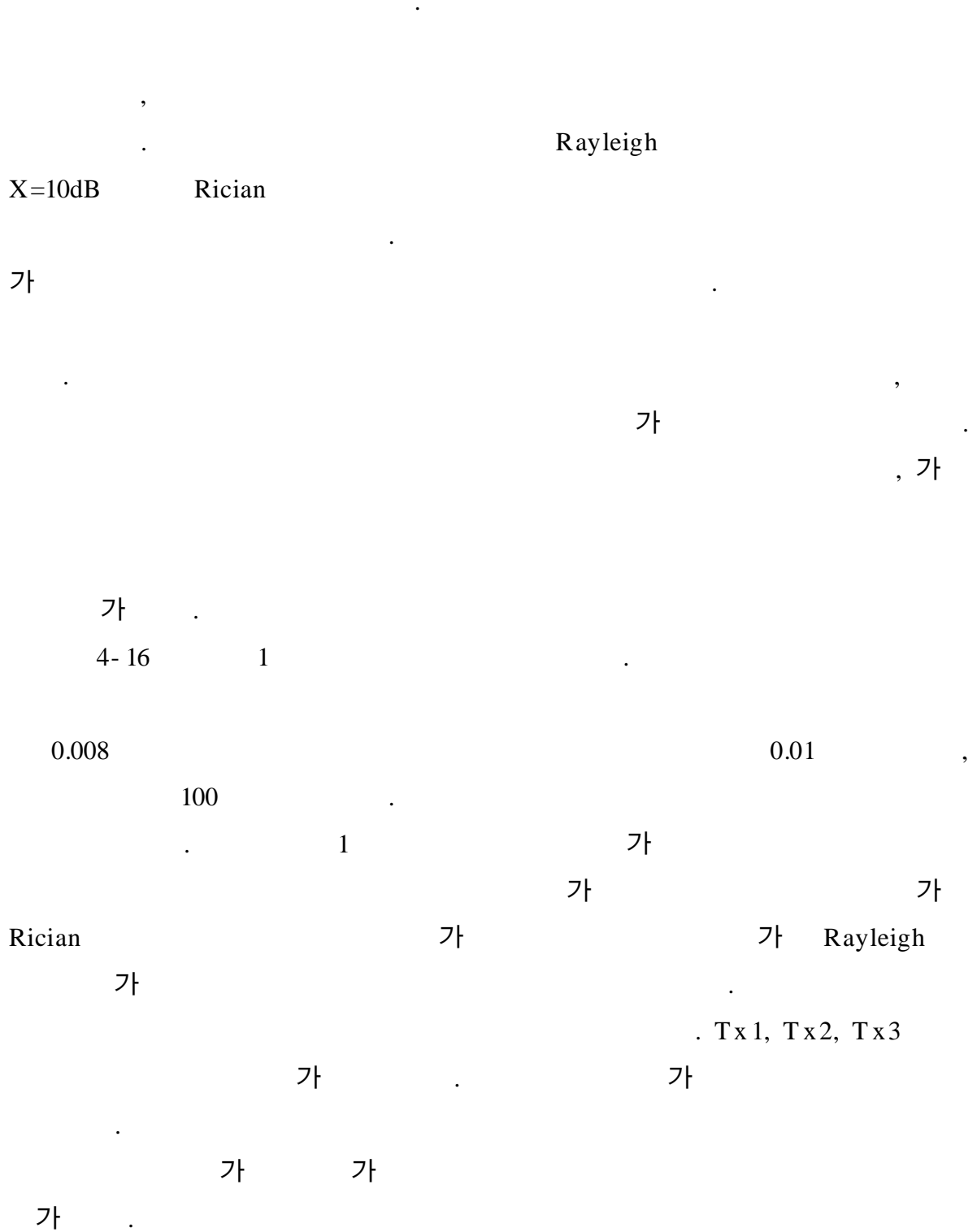
(a)

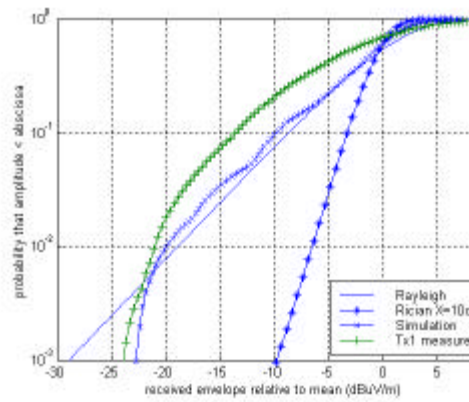




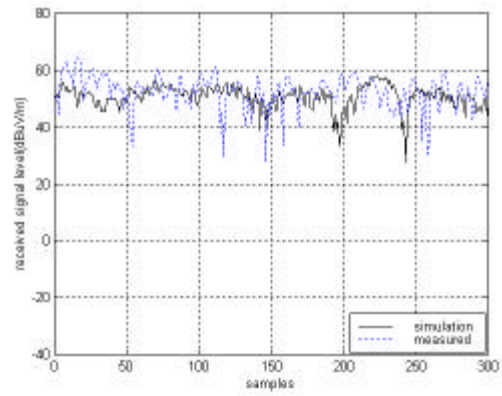
(a)



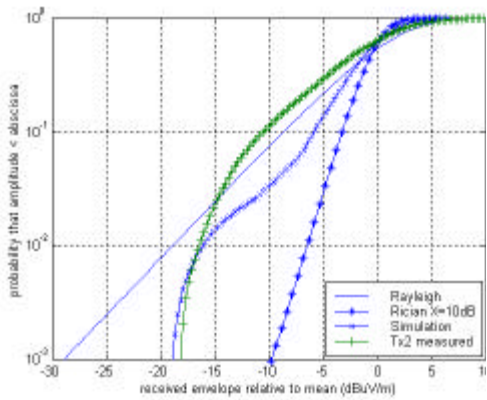




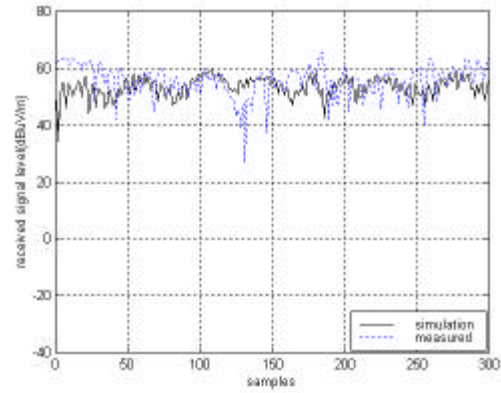
(a) Tx 1



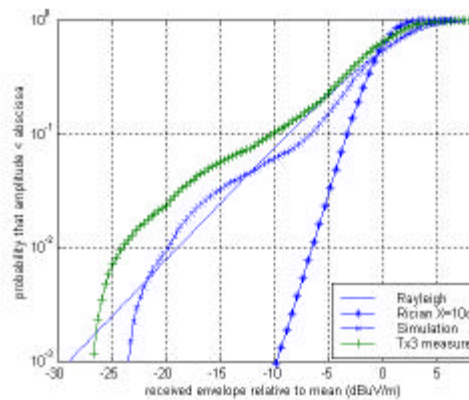
(b) Tx 1



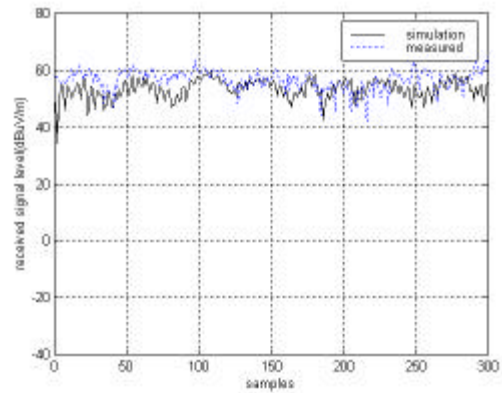
(c) Tx 2



(d) Tx 2

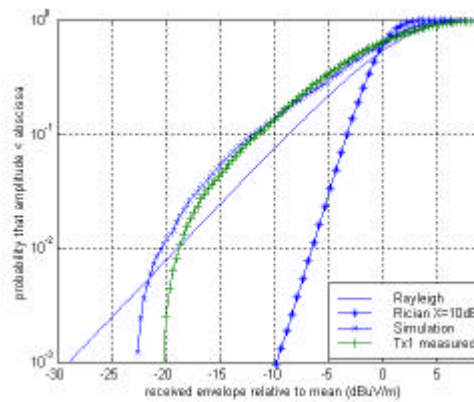


(e) Tx 3

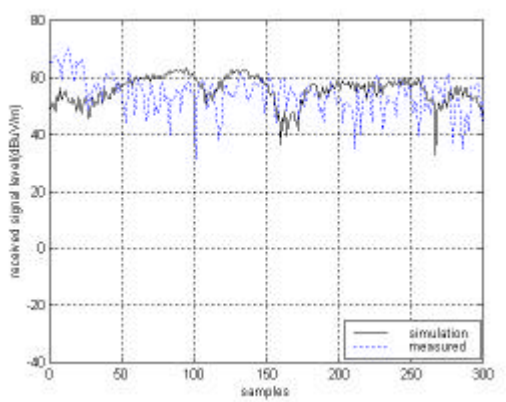


(f) Tx 3

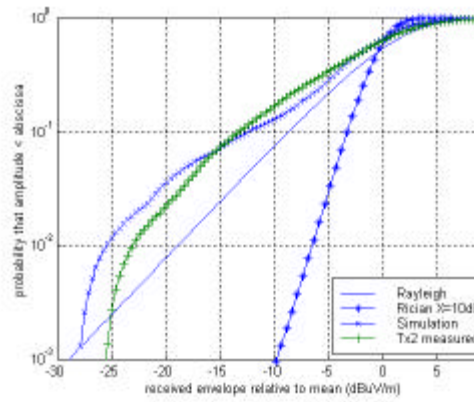
4- 17. 2 가 0.001 , Tx3 Tx4 가 Tx4 가 가 .



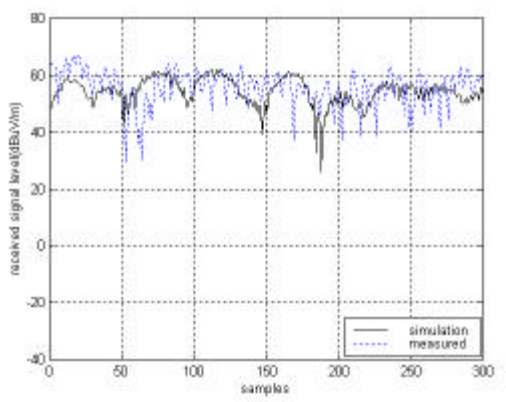
(a) Tx 1



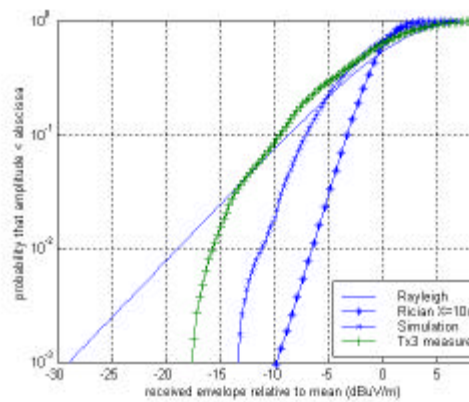
(b) Tx 1



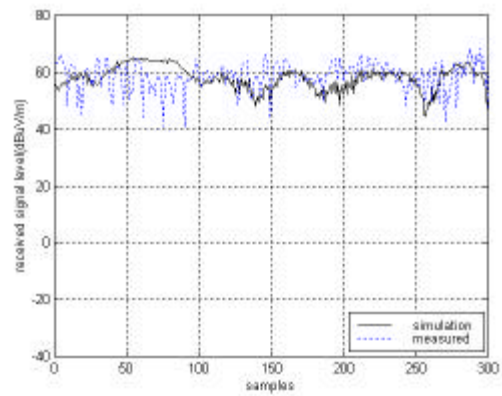
(c) Tx 2



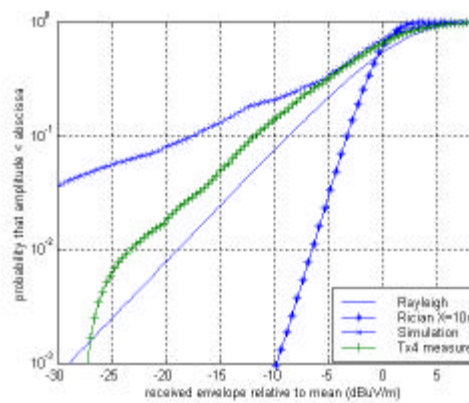
(d) Tx 2



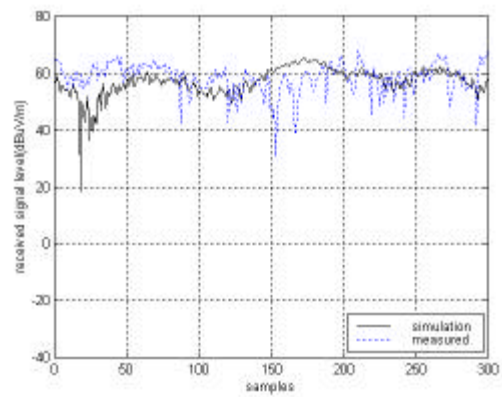
(e) Tx3



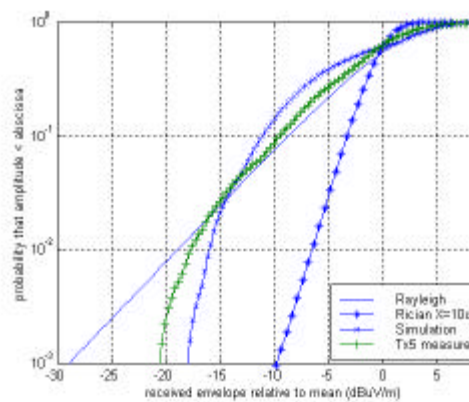
(f) Tx3



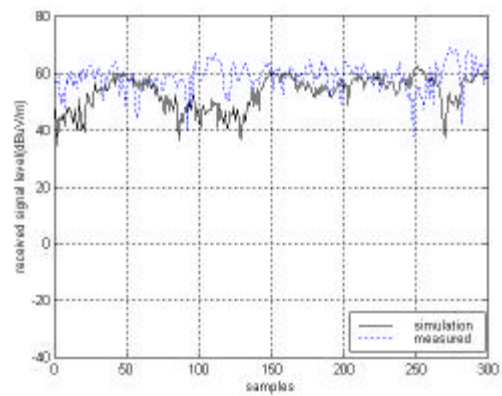
(g) Tx4



(h) Tx4



(i) Tx5



(j) Tx5

가 ,

Tx3

X=10dB Rician 가 , 가

가

. Tx4 Tx5 Tx5가 Rician 가

Tx4 가 Tx5

, Tx1, Tx2, Tx3

가 Tx4가 가 .

가 .

2

.

4- 18 3 .

0.01

, 100 .

, 3 가 가

, Rayleigh

.

3 2 가

Rayleigh 가 3 Rayleigh

가 가

.

4- 19 4 . 4

가

0.2

.

, 4

X=10dB Rician Rician

4- 19

.

1

가

, 가

0.01

Rayleigh

.

,

.

2

가

가

0.001

.

1

가

가

.

.

.

3

0.01

, 1 2

1 2

가

Rayleigh

.

4

0.2

.

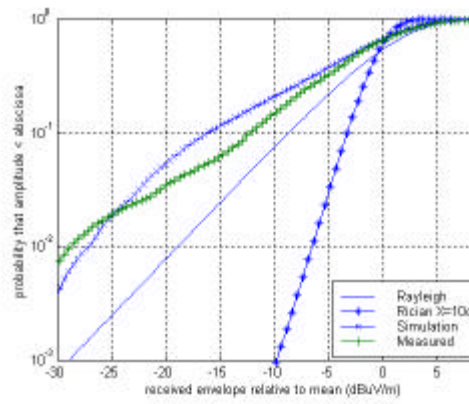
.

가

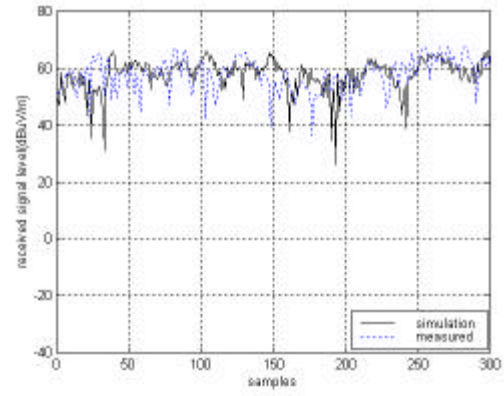
가

,

.

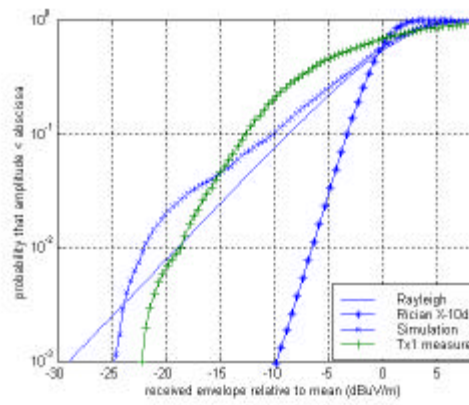


(a)

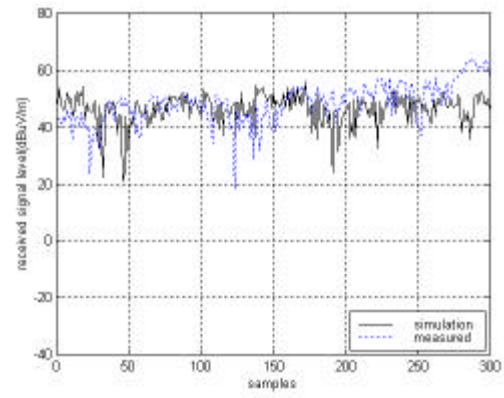


(b)

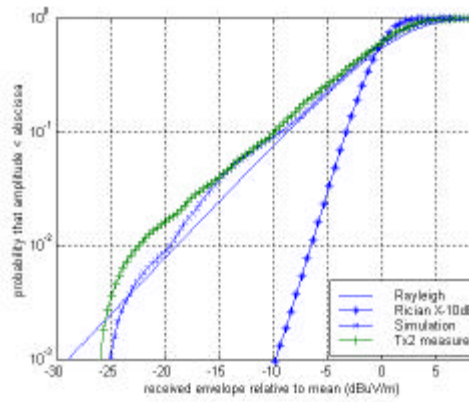
4- 18. 3



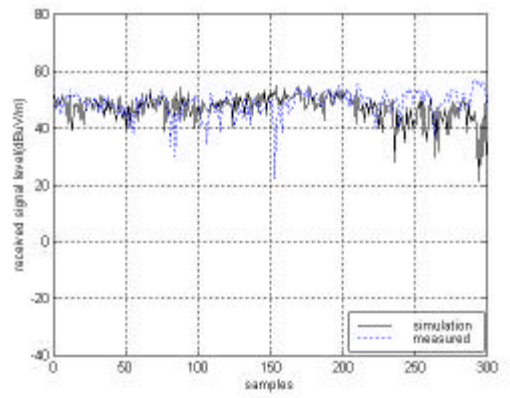
(a) Tx 1



(b) Tx 1



(c) Tx 2



(d) Tx 2

4- 19. 4

RCS

RCS

bistatic RCS

RCS

RCS

100

, RCS

가

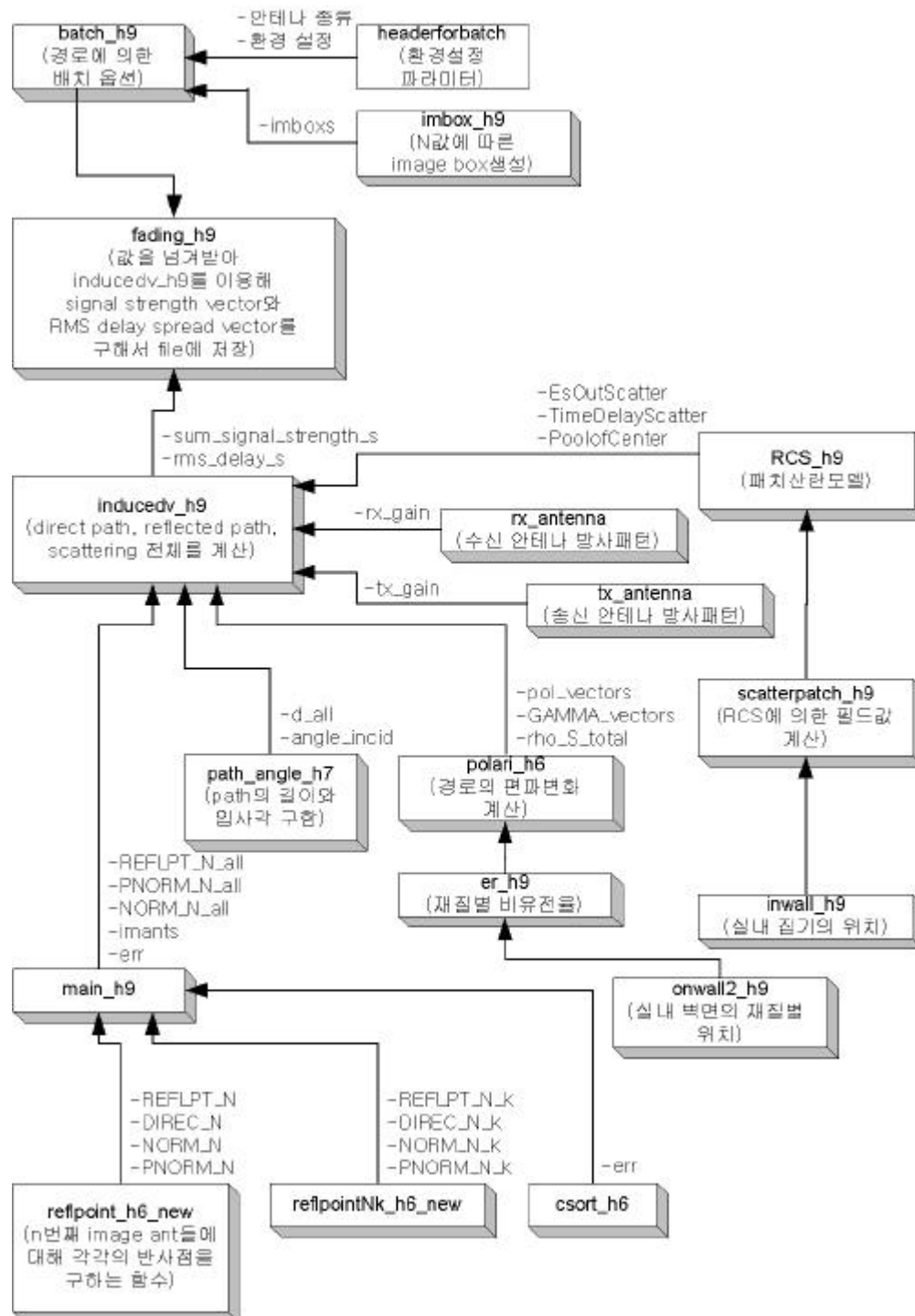
4

RCS

- [1] Manuel F. Catedra, *Cell Planning for Wireless Communications*, Artech House Inc., 1999.
- [2] G.E. Athanasiadou, J.P. McGeehan, "A new 3D Indoor ray-Tracing Propagation Model with particular reference to the prediction of power and delay spread ," *Proceedings of the 6th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications - Volume 3*, pp.1161- 1165
- [3] Driessen, P.F. "Development of a propagation model in the 20-60 GHz band for wireless communications," *Proc. IEEE Pac. Rim Conf. on Comm., Comp., Sig. Proc.* May 1991, pp. 59-62
- [4] G.T. Martin, M. Faulkner, M.A. Beach, "Wide Band Propagation Measurements and Ray Tracing Simulations at 1890 MHz," *Proceedings of the 1995 4th IEEE International Conference on Universal personal Communications*, pp.283- 287
- [5] Giovanni Emanuel Corazza, "A Characterization of Indoor Space and Frequency Diversity by Ray-Tracing Modeling," *Journal on Selected Areas in Communications*, Vol.14, No.3, pp.411-419, 1996
- [6] C.A. Balanis, *Antenna Theory Analysis and Design 2nd edition*, John Wiley & Sons, INC., 1997
- [7] D.M. Pozar, *Microwave Engineering 2nd edition*, John Wiley & Sons, INC., 1998
- [8] E. F. Knott, D. J. Cichon and W. Wisebeck, *Radar Cross Section 2nd*, Norwood, MA, Artech House, 1993
- [9] G. T. Ruck, D. E. Barrick, W. D. Stuart and C. K. Krichbaum, *Radar Cross Section Handbook*. New York, Plenum, 1970
- [10] Tarng J.H, K.M. Ju, "A novel 3-D scattering model of 1.8-GHz radio propagation in microcellular urban environment," *IEEE Trans. on Electromagnetic Compatibility*, Vol. 41 No. 2, pp.100- 106, 1999

- [11] R.G. Vaughan, " Polarization diversity in mobile communication," *IEEE Trans. Veh. Technol*, Vol. VT - 39, No. 3 pp. 177- 186, 1990
- [12] S.Y. Tan, H.S. Tan, "A Microcellular Communications Propagation Model Based on the Uniform Theory of Diffraction and Multiple Image Theory," *IEEE Trans. on Antenna and Propagation*, Vol. 44 No. 10, pp.1317- 1326, October 1996
- [13] Kazimierz Siwiak, *Radio Propagation and Antennas for Personal Communication*, Artech House Inc., 1995
- [14] Kaveh Pahlavan, Allen H. Levesque, *Wireless Information networks*, John Wiley & Sons Inc., 1995
- [15] Constantine A. Balanis, *Advanced Engineering Electromagnetics*, John Wiley & Sons Inc., 1989
- [16] K.G. Budden, *The Propagation of radio waves*, Cambridge University Press, 1985
- [17] JinMan Kim, SunHak Hong, and YoungJoong Yoon, "A Study on the Space & Frequency Diversity in Microcell Using Ray-tracing," *ITC- CS CC '98, Sokcho*, Vol. 1, pp. 813- 816, 1998. 7.
- [18] SunHak Hong, YoungSu Lee, MyungSun Choi* and YoungJoong Yoon, "Polarization Diversity with Directional Antennas for Indoor Environments," *APM C99*
- [19] , , , "3
," , 10 5
, 1999, 9
- [20] , , , , , "
," 1999 ('99.
11. 20.:), pp. 817- 820

Line of Sight MATLAB source code



block diagram

<<headerforbatch.m>>

```
%
                                LOS
N=5;                                %reflection order

f=1.89*10^9;

numofsample=300;

ROOMSIZE=[21.65 8 2.88];           %room size

RXPOINT_pool(1,:) = [0.93 7.25 2.44];           %Rx antenna location Rx 1

x=19.1;
y_start=7.55;
y_end=0.95;
tx_height=1.67;
ROUTE_pool(:,1)=[x*ones(1,numofsample)' linspace(y_start, y_end, numofsample)' tx_height*ones(1,numofsample)'];

x=17.3;
y_start=7.55;
y_end=0.95;
tx_height=1.67;
ROUTE_pool(:,2)=[x*ones(1,numofsample)' linspace(y_start, y_end, numofsample)' tx_height*ones(1,numofsample)'];

x_start=19.77;
x_end=8.37;
y=6.95;
tx_height=1.67;
ROUTE_pool(:,3)=[linspace(x_start, x_end, numofsample)' y*ones(1,numofsample)' tx_height*ones(1,numofsample)'];
```

<<imbox_h9.m>>

```
function imboxs=imbox_h9(N)
% imboxs=imbox_h9(N)
%
% Return : relative location of image boxes

% File Log - SunHak Hong
% 11/01/99 - File Updated

imboxs=[];
for index1= -N:N
    for index2= -N:N
        for index3= -N:N

            if ( abs(index1)+abs(index2)+abs(index3) <=N )&( abs(index1)+abs(index2)+abs(index3) ~=0 )

                imboxs=[imboxs ; index1 index2 index3];
            end
        end
    end
end
```

[illegible]

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RX_LOCA_MENU=1:2;
ROUTE_MENU=1:1;
POL_RX_MENU=1:4;
RX_ANTENNA_MENU=2:5;

for rx_loca_menu=RX_LOCA_MENU
    for route_menu=ROUTE_MENU

        FILENAMES=[];

        for rx_antenna_menu=RX_ANTENNA_MENU
            for pol_rx_menu=POL_RX_MENU

                rx_loca='a1'*(rx_loca_menu==1) + 'a2'*(rx_loca_menu==2);
                route='u1'*(route_menu==1)+'u2'*(route_menu==2)+'u3'*(route_menu==3)+'u4'*(route_menu==4);

                rx_antenna='i'*(rx_antenna_menu==1)+'dd'*(rx_antenna_menu==2)+'m1'*(rx_antenna_menu==3)+'m_ '*(rx_antenna_me
                nu==4)+'m4'*(rx_antenna_menu==5);

                pol_rx='v'*(pol_rx_menu==1)+'h'*(pol_rx_menu==2)+'l'*(pol_rx_menu==3)+'r'*(pol_rx_menu==4);
                if ~(((rx_antenna_menu==3)|(rx_antenna_menu==4))&((pol_rx_menu==3)|(pol_rx_menu==4))) % omission
of pol45 diversity at 1x4 %this line is also in inducedv_h9
                    filename=str2mat([rx_loca route rx_antenna pol_rx]);
                    FILENAMES=[FILENAMES ;filename]; % 14 files ---> 12 files
                end
            end
        end
    end

    RXPOINT=RXPOINT_pool(rx_loca_menu,:);
    ROUTE=ROUTE_pool(:,route_menu);
    fading_h9(FILENAMES,new_folder,RXPOINT,ROUTE,RX_ANTENNA_MENU);

end
end

```

<<fafing_h9.m>>

```
function [streng,rmsdelay]=fading_h9(FILENAMES,new_folder,RXPOINT,ROUTE,RX_ANTENNA_MENU)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
global N
```

```
global imboxs
```

```
global ROOMSIZE
```

```
global f
```

```
global lambda
```

```
global beta_a
```

```

global numofsample
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%global Pol_Compen_Factor
%Pol_Compen_Factor=1;
%numofsample=300;

%global RXPOINT
%RXPOINT=[2 0.37 1.3]; %rx 1

%global pol_tx
pol_err=0.1;
pol_tx=([0 0 1]+pol_err)/sqrt(sum([0 0 1]+pol_err));
%pol_tx=([1/sqrt(2) 0 1/sqrt(2)]+pol_err)/sqrt(sum([1/sqrt(2) 0 1/sqrt(2)]+pol_err)); %45Deg inclined

%pol_rx_menu=menu('Polarization of Rx','vertical','Horizontal','+45','-45')
%POL_RX=[0 0 1];
POL_RX=[0 0 1 ; 0 1 0 ; 0 1/sqrt(2) 1/sqrt(2) ; 0 -1/sqrt(2) 1/sqrt(2) ];

%Tx_ant dipole
tx_antenna_menu=2;

% [filename,pname1]=uinputfile('*.mat', 'Input the file name to save');
% dataname=filename(1:end-4);

% dataname=filename;
% file_name=sprintf('filename=%s',filename);

fprintf('\n\n')
disp('index_number')
[num_of_data,C_length]=size(FILENAMES);

STRENG=zeros(numofsample,num_of_data);
RMSDELAY=zeros(numofsample,num_of_data);
for index=1:numofsample
    disp(FILENAMES(1,1:4));
    % index_sample=sprintf('Index_Of_Sample=%d',index);

    disp(index);

    TXPOINT=ROUTE(index,:);
    %
    [sum_signal_strength_s,rms_delay_s,theta_all,phi_all,path_length_all,delay_all,V_all_dB]=inducedv_h9(TXPOINT,RXPOIN
T,pol_tx,POL_RX,RX_ANTENNA_MENU,tx_antenna_menu);

    [sum_signal_strength_s,rms_delay_s]=inducedv_h9(TXPOINT,RXPOINT,pol_tx,POL_RX,RX_ANTENNA_MENU,tx_ant
enna_menu);

```

```

% Show option !!!!!!!
% a1=sprintf('rms_delay=%0.5g ns',rms_delay*10^9);
% disp(a1);
% a2=sprintf('sum_signal_strength=%0.5g\n\n',sum_signal_strength);
% disp(a2);

STRENG(index,:)=20*log10(sum_signal_strength_s)+100;
RMSDELAY(index,:)=rms_delay_s;
end
clear index
clear sum_signal_strength_s rms_delay_s

filename_vectors=FILENAMES;
rmsdelay_vectors=RMSDELAY;
streng_vectors=STRENG;

% streng_vectors=scaling(FILENAMES,STRENG);

cd(new_folder)

new_file=FILENAMES(1,1:4);
%eval(['save ' new_file ' filename_vectors rmsdelay_vectors streng_vectors theta_all phi_all path_length_all delay_all
V_all_dB'])
eval(['save ' new_file ' filename_vectors rmsdelay_vectors streng_vectors '])

% save data2 filename_vectors rmsdelay_vectors streng_vectors
cd('..')

lcr_h9(new_file,new_folder)
%correl_h9(new_file,new_folder)

% sub-function
% % % % % % % %
function streng_vectors=scaling(FILENAMES,STRENG)

%eval(['load ' FILENAMES(1,1:4) '_mstd MeasuredFileName MeasuredData MeasuredMean MeasuredStd'])

[ row_x,col_x]=size(STRENG)
for index=1:col_x
    TEMP=STRENG(:,index);
    TempMean(index)=mean(TEMP);
    TempStd(index)=std(TEMP);
end
MeanAverage=mean(TempMean);
StdAverage=mean(TempStd);

for index=1:col_x
    TEMP=STRENG(:,index);
    %TEMP=(TEMP - mean(TEMP))/std(TEMP);
    TEMP=(TEMP - MeanAverage)/StdAverage;
    %TEMP=(TEMP+MeasuredMean(index))*MeasuredStd(index);
    %TEMP=TEMP*4.9235+43.4525; %

```



```

TEMP=TEMP*4.9235+43.4525;
streng_vectors(:,index)=TEMP;
end

```

<<inducedv_h9.m>>

```

function
[sum_signal_strength_s,rms_delay_s,theta_all,phi_all,path_length_all,delay_all,V_all_dB]=inducedv_h9(TXPOINT,RXPOINT,
T,pol_tx,POL_RX,RX_ANTENNA_MENU,tx_antenna_menu)
%
%POL_RX, RX_ANTENNA_MENU          loop

```

```

global N

```

```

global ROOMSIZE

```

```

[REFLPT_N_all,PNORM_N_all,NORMDI_N_all,imants,err]=main_h6(TXPOINT,RXPOINT);

```

```

[d_all,angle_incid]=path_angle_h7(TXPOINT,RXPOINT,REFLPT_N_all,PNORM_N_all,NORMDI_N_all,imants,err);

```

```

global f

```

```

global lambda

```

```

global beta_a

```

```

%%%%%%%%%%
REFLPT_N_all=REFLPT_N_all(end:-1:1,:);
PNORM_N_all=PNORM_N_all(end:-1:1,:);
NORMDI_N_all=NORMDI_N_all(end:-1:1,:);
angle_incid=angle_incid(end:-1:1,:);
%%%%%%%%%%

```

```

[pol_vectors,GAMMA_vectors,rho_s_total]=polari_h6(REFLPT_N_all,PNORM_N_all,NORMDI_N_all,pol_tx,angle_incid);

```

```

%global Pol_Compen_Factor

```

```

global theta_tilting

```

```

global phi_tilting

```

```

%theta_tilting=acos(pol_rx(3));
%phi_tilting=atan(pol_rx(1)/(pol_rx(2)+eps));

```

```

global mini_delay

```

```

global maxi_delay

```

```

patch_on=1;%1

```

```

%delayspread!!(Delay Spread      )
%theta=[];
%phi=[];
%path_length_all=[];
%%

```

```

%direct path
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dR_T=RXPOINT - TXPOINT;
dT_R=TXPOINT - RXPOINT;
rd=sqrt(sum((dR_T).^2));

thetaR_T=acos(dR_T(3)/rd);
%%%phi    2pi          71
if dR_T(1)>=0 & dR_T(2)>=0
    phiR_T=atan(dR_T(2)/dR_T(1));
elseif dR_T(1)>=0 & dR_T(2)<0
    phiR_T=atan(dR_T(2)/dR_T(1));
elseif dR_T(1)<0 & dR_T(2)>=0
    phiR_T=acos(dR_T(1)/sqrt(dR_T(1)^2+dR_T(2)^2));
else
    phiR_T=atan(dR_T(2)/dR_T(1))-pi;
end

thetaT_R=acos(dT_R(3)/rd);
if dT_R(1)>=0 & dT_R(2)>=0
    phiT_R=atan(dT_R(2)/dT_R(1));
elseif dT_R(1)>=0 & dT_R(2)<0
    phiT_R=atan(dT_R(2)/dT_R(1));
elseif dT_R(1)<0 & dT_R(2)>=0
    phiT_R=acos(dT_R(1)/sqrt(dT_R(1)^2+dT_R(2)^2));
else
    phiT_R=atan(dT_R(2)/dT_R(1))-pi;
end

pathloss=(1/rd) * exp(-i*beta_a*rd) ;
delay_d=d/(3*10^8);

%for Tx Antenna
DirectionTx=TXPOINT.*[0 1 1]; %to fix the direction of antenna %Rx (OmniDirectional)
)
dP_T=DirectionTx - TXPOINT;
thetaP_T=acos(dP_T(3)/sqrt(sum((dP_T).^2)));
if dP_T(1)>=0 & dP_T(2)>=0
    phiP_T=atan(dP_T(2)/dP_T(1));
elseif dP_T(1)>=0 & dP_T(2)<0
    phiP_T=atan(dP_T(2)/dP_T(1));
elseif dP_T(1)<0 & dP_T(2)>=0
    phiP_T=acos(dP_T(1)/sqrt(dP_T(1)^2+dP_T(2)^2));
else
    phiP_T=atan(dP_T(2)/dP_T(1))-pi;
end

tx_gain=tx_antenna(thetaR_T,phiR_T,thetaP_T,phiP_T,tx_antenna_menu);

%for Rx Antenna
DirectionRx=ROOMSIZE.*[1 0 0]+RXPOINT.*[0 0 1]; %to fix the direction of antenna( ~!

```

```

~l)
dP_R=DirectionRx - RXPOINT ;
thetaP_R=acos(dP_R(3)/sqrt(sum((dP_R).^2)));
if dP_R(1)>=0 & dP_R(2)>=0
    phiP_R=atan(dP_R(2)/dP_R(1));
elseif dP_R(1)>=0 & dP_R(2)<0
    phiP_R=atan(dP_R(2)/dP_R(1));
elseif dP_R(1)<0 & dP_R(2)>=0
    phiP_R=acos(dP_R(1)/sqrt(dP_R(1)^2+dP_R(2)^2));
else
    phiP_R=atan(dP_R(2)/dP_R(1))-pi;
end

%%%DelaySpread(DelaySpread
)
%theta=[theta; thetaT_R - thetaP_R+pi/2]; %NEW
%phi=[phi; phiT_R - phiP_R]; %NEW;
%disp('phi')
%disp(phi*(180/pi))
%path_length_all=[path_length_all; rd];
%%%

%%TestRCS : patch scattered fields
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if patch_on==1
[EsOutScatter,TimeDelayScatter,PoolofCenter]=RCS_h9(TXPOINT,RXPOINT,pol_tx,tx_antenna_menu); %before
considering Rx considering
EsOutScatter=EsOutScatter;

for index_3=1:length(TimeDelayScatter)
    Center=PoolofCenter(index_3,:);
    RayDirectionJtoRx=RXPOINT - Center;
    theta_JtoRx(index_3,1)=acos(RayDirectionJtoRx(3)/norm(RayDirectionJtoRx));
    if RayDirectionJtoRx(1)>=0 & RayDirectionJtoRx(2)>=0
        phi_JtoRx(index_3,1)=atan(RayDirectionJtoRx(2)/RayDirectionJtoRx(1));
    elseif RayDirectionJtoRx(1)>=0 & RayDirectionJtoRx(2)<0
        phi_JtoRx(index_3,1)=atan(RayDirectionJtoRx(2)/RayDirectionJtoRx(1));
    elseif RayDirectionJtoRx(1)<0 & RayDirectionJtoRx(2)>=0
        phi_JtoRx(index_3,1)=acos(RayDirectionJtoRx(1)/sqrt(RayDirectionJtoRx(1)^2+RayDirectionJtoRx(2)^2));
    else
        phi_JtoRx(index_3,1)=atan(RayDirectionJtoRx(2)/RayDirectionJtoRx(1))-pi;
    end
end

end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
V_d=[]; EsOutScatterS=[];
for index_1=2:(length(RX_ANTENNA_MENU)+1)

```

```

rx_antenna_menu=RX_ANTENNA_MENU(index_1- 1);
rx_gain=rx_antenna(thetaT_R,phiT_R,thetaP_R,phiP_R,rx_antenna_menu);

%%for Rx Antenna of Scattering
G_rj=rx_antenna(thetaJtoRx,phiJtoRx,thetaP_R,phiP_R,rx_antenna_menu);

for index_2=1:size(POL_RX,1)    %%
    if ~(((rx_antenna_menu==3)|(rx_antenna_menu==4))&((index_2==3)|(index_2==4)))
        V_d_temp=tx_gain * rx_gain * pathloss * sum(pol_tx.*POL_RX(index_2,:));
        V_d=[V_d V_d_temp];

        %for scattering
        if patch_on==1
            EsOutScatterTEMP=sum( (EsOutScatter.*repmat(POL_RX(index_2,:),length(TimeDelayScatter),1) )')' .*
G_rj;

            EsOutScatterS=[EsOutScatterS EsOutScatterTEMP];
        end
    end

end

end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%reflected path
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
TEMP_B=triu(repmat(4*(1:N).^2 +2,N,1));
TEMP_C=cumsum(TEMP_B)';
TEMP_D=TEMP_C(:,N);
TEMP_E=cumsum([0 ; TEMP_D(1:end- 1)]);
TEMP_F=triu(TEMP_C+repmat(TEMP_E,1,N));
TEMP_G=triu(TEMP_F- TEMP_B+1);
index_start=triu(TEMP_G);
index_length=TEMP_B;
index_end=TEMP_F;

delay_r_all=[];
V_r_all=[];
angle_azi_r_all=[];
angle_ele_r_all=[];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for index=1:N
    index_1=index_start(index,index);
    index_2=index_end(index,index);
    reflpt_1=REFLPT_N_all(index_start(1,index):index_end(1,index),:);
    reflpt_N=REFLPT_N_all(index_1:index_2,:);
    pol_N=pol_vectors(index_1:index_2,:);
    % GAMMA_N=GAMMA_vectors(index_1:index_2,:);
    % GAMMA
    START =index_start(:,index);
    START =START (find(START ~=0));
    END=index_end(:,index);
    END=END(find(END~=0));
    GAMMA_N=ones(index_length(1,index),1);

    for index_G=1:index

```

```

        GAMMA_n(:,index_G)=GAMMA_vectors(START(index_G):END(index_G));
        GAMMA_N=GAMMA_N.*GAMMA_n(:,end);
end

%scattering
rho=rho_s_total(index_1:index_2,:);
dIR_T=reflpt_1-repmat(TXPOINT,index_length(index,index),1); %NEW
dIT_R=reflpt_N-repmat(RXPOINT,index_length(index,index),1); %NEW
%dIR_T=repmat(RXPOINT,index_length(index),1)-reflpt_N;
%dIT_R=reflpt_N-repmat(RXPOINT,index_length(index),1);

rr1=(sqrt(sum((dIR_T).^2)))/rr1;
rr2=(sqrt(sum((dIT_R).^2)))/rr2;

if index==1
    path_length=d_all(1:index_length(index,index))';
else
    %sum(index_length(1:index-1,index-1))+1
    %sum(index_length(1:index,index))
    %size(d_all)
    path_length=d_all(index_start(1,index):index_end(1,index))';
end

thetaIR_T=acos(dIR_T(:,3)/rr1);
%% %phi    2pi    ~!!
phiIR_T=[]; %

phiIR_T_temp1=dIR_T(:,1);
phiIR_T_temp2=dIR_T(:,2);

plane1=find(phiIR_T_temp1>=0 & phiIR_T_temp2>=0);
phiIR_T(plane1)=atan(phiIR_T_temp2(plane1)/phiIR_T_temp1(plane1));

plane2=find(phiIR_T_temp1>=0 & phiIR_T_temp2<0);
phiIR_T(plane2)=atan(phiIR_T_temp2(plane2)/phiIR_T_temp1(plane2));

plane3=find(phiIR_T_temp1<0 & phiIR_T_temp2>=0);
phiIR_T(plane3)=acos(phiIR_T_temp1(plane3)/sqrt(phiIR_T_temp1(plane3).^2+phiIR_T_temp2(plane3).^2))+pi/2;

plane4=find(phiIR_T_temp1<0 & phiIR_T_temp2<0);
phiIR_T(plane4)=atan(phiIR_T_temp2(plane4)/phiIR_T_temp1(plane4))-pi;

phiIR_T_n=(phiIR_T)';
%%

thetaIT_R=acos(dIT_R(:,3)/rr2);

%% %phi    2pi    index
phiIT_R=[]; %

phiIT_R_temp1=dIT_R(:,1);
phiIT_R_temp2=dIT_R(:,2);

```

```

plane_1=find(phiIT_R_temp1>=0 & phiIT_R_temp2>=0);
phiIT_R(plane_1)=atan(phiIT_R_temp2(plane_1)/phiIT_R_temp1(plane_1));

plane_2=find(phiIT_R_temp1>=0 & phiIT_R_temp2<0);
phiIT_R(plane_2)=atan(phiIT_R_temp2(plane_2)/phiIT_R_temp1(plane_2));

plane_3=find(phiIT_R_temp1<0 & phiIT_R_temp2>=0);
phiIT_R(plane_3)=acos(phiIT_R_temp1(plane_3)/sqrt(phiIT_R_temp1(plane_3).^2+phiIT_R_temp2(plane_3).^2));

plane_4=find(phiIT_R_temp1<0 & phiIT_R_temp2<0);
phiIT_R(plane_4)=atan(phiIT_R_temp2(plane_4)/phiIT_R_temp1(plane_4))-pi;

phiIT_R_n=(phiIT_R)';

%%%NaN
if length(phiIT_R_n)~=length(thetaIT_R)
    phiIT_R_n=[phiIT_R_n; repmat(NaN,abs(length(phiIT_R_n)-length(thetaIT_R)),1)];
end

%%%
~!

%%%Delay Spread
%theta=[theta; thetaIT_R-thetaP_R+pi/2]; %NEW
%phi=[phi; phiIT_R_n-phiP_R]; %NEW;
%path_length_all=[path_length_all; path_length];
%%%

pathloss_r=(1/path_length) .*exp(-i*beta_a*path_length) ;
delay_r=path_length/(3*10^8);
delay_r_all=[delay_r_all ; delay_r];
angle_azi_r_all=[angle_azi_r_all ; phiIT_R_n];
angle_ele_r_all=[angle_ele_r_all ; thetaIT_R];

tx_gain_r=tx_antenna(thetaIR_T,phiIR_T,thetaP_T,phiP_T,tx_antenna_menu);
V_r=[];
for index_1=2:(length(RX_ANTENNA_MENU)+1) %antenna
    rx_antenna_menu=RX_ANTENNA_MENU(index_1-1);

    rx_gain_r=rx_antenna(thetaIT_R,phiIT_R_n,thetaP_R,phiP_R,rx_antenna_menu);

    for index_2=1:size(POL_RX,1) %polarization
        if ~(((rx_antenna_menu==3)|(rx_antenna_menu==4))&((index_2==3)|(index_2==4))) %omission of pol45
            diversity at 1x4 %this line is also in batch_h9
            %%% - - >here
            V_r_temp=tx_gain_r .* rx_gain_r .* pathloss_r .* sum( (pol_N .*
repmat(POL_RX(index_2,:),index_length(index,index),1))' )' .*rho .*GAMMA_N;
            V_r=[V_r V_r_temp];
        end
    end
end
end

```

```

clear GAMMA_n GAMMA_N
V_r_all=[V_r_all ; V_r];

end
clear index index_1 index_2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

delay_all=[delay_d ; delay_r_all];
V_all=[V_d ; V_r_all];

%%DelaySpread
% V_all_dB=20*log10(abs(V_all));
% theta_all=theta.*(180/pi);
% phi_all=phi.*(180/pi);
% % %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if patch_on==1
delay_all=[delay_all; TimeDelayScatter];
V_all=[V_all; EsOutScatterS];
% V_all=V_all/(1+index_end(1,N)+length(TimeDelayScatter)); %normalize
end

V_all=V_all;%(1+index_end(1,N)); %normalize

[time_arrival,index_arrival]=sort(delay_all);
induced_voltage(1:length(index_arrival),:)=V_all(index_arrival(1:length(index_arrival),1),:);

% angle_azi_all=[thetaT_R ; angle_azi_r_all];
% angle_ele_all=[phiT_R ; angle_ele_r_all];
% azi_angle(1:length(index_arrival),1)=angle_azi_all(index_arrival(1:length(index_arrival),1));
% ele_angle(1:length(index_arrival),1)=angle_ele_all(index_arrival(1:length(index_arrival),1));

[index_rms_2,kind_ant]=find( 20*log10(induced_voltage/ repmat(max(induced_voltage),length(induced_voltage),1))
>=-50 );

[L_length,C_length]=size(induced_voltage);
rms_delay=[];
sum_signal_strength=[];
for index_column=1:C_length
    mean_delay=sum(abs(induced_voltage(index_rms_2,index_column)).*time_arrival(index_rms_2)) /
sum(abs(induced_voltage(index_rms_2,index_column)));
    second_moment=sum(abs(induced_voltage(index_rms_2,index_column)).*time_arrival(index_rms_2).^2) /
sum(abs(induced_voltage(index_rms_2,index_column)));
    rms_delay_temp=sqrt(second_moment-mean_delay.^2);
    rms_delay=[rms_delay rms_delay_temp];

    induced_voltage_temp=induced_voltage(:,index_column);

    sum_signal_strength_temp=abs(sum(induced_voltage_temp(~isnan(induced_voltage_temp)).*exp(-i*2*pi*f*time_arrival(~
isnan(induced_voltage_temp)))) );
    sum_signal_strength=[sum_signal_strength sum_signal_strength_temp];
end

```

```

rms_delay_s=rms_delay;
sum_signal_strength_s=sum_signal_strength/index_end(1,N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

<<main_h6.m>>

function [REFLPT_N_all,PNORM_N_all,NORMDI_N_all,imants,err]=main_h6(TXPOINT,RXPOINT)

global ROOMSIZE
global N
global imboxs

%run pre-required programs
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
imants=2*(-(imboxs<0)+(imboxs>=0)).*repmat(ROOMSIZE,length(imboxs),1).*fix(abs(imboxs+(imboxs>=0))/2)...
    + repmat(TXPOINT,length(imboxs),1).*(-1).^(abs(imboxs));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[REFLPT_N,DIREC_N,NORMDI_N,PNORM_N]=reflpoint_h6_new(imants,TXPOINT,RXPOINT);

REFLPT_N_all=REFLPT_N;
DIREC_N_all=DIREC_N;
NORMDI_N_all=NORMDI_N;
PNORM_N_all=PNORM_N;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
TEMP_B=triu(repmat(4*(1:N).^2+2,N,1));
TEMP_C=cumsum(TEMP_B');
TEMP_D=TEMP_C(:,N);
TEMP_E=cumsum([0;TEMP_D(1:end-1)]);
TEMP_F=triu(TEMP_C+repmat(TEMP_E,1,N));
TEMP_G=triu(TEMP_F-TEMP_B+1);
index_start=triu(TEMP_G);
index_length=TEMP_B;
index_end=TEMP_F;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for index=2:N
    %a=index_s(index-1);
    a=index_start(index-1,index);
    %b=length(REFLPT_N_all)
    b=index_end(index-1,N);

    REFLPT_N_k_1=REFLPT_N_all(a:b,:);
    DIREC_N_k_1=DIREC_N_all(a:b,:);
    NORMDI_N_k_1=NORMDI_N_all(a:b,:);
    PNORM_N_k_1=PNORM_N_all(a:b,:);
end
```



```

[REFLPT_N_k,DIREC_N_k,NORMDI_N_k,PNORM_N_k] ...
    =reflpointNk_h6_new(imants,TXPOINT,RXPOINT,REFLPT_N_k_1,DIREC_N_k_1,NORMDI_N_k_1,index);

    REFLPT_N_all=[REFLPT_N_all ; REFLPT_N_k];
    DIREC_N_all=[DIREC_N_all ; DIREC_N_k];
    NORMDI_N_all=[NORMDI_N_all ; NORMDI_N_k];
    PNORM_N_all=[PNORM_N_all ; PNORM_N_k];
end
clear a b

%determine the error locations
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
e1=find(REFLPT_N_all(:,1)<0 | REFLPT_N_all(:,1)>ROOMSIZE(1));
e2=find(REFLPT_N_all(:,2)<0 | REFLPT_N_all(:,2)>ROOMSIZE(2));
e3=find(REFLPT_N_all(:,3)<0 | REFLPT_N_all(:,3)>ROOMSIZE(3));

err=csort_h6([e1; e2; e3]);
fprintf('location of error \n')
fprintf('\t%.0f \n',err(1,:));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%make error REFLPT_N_all NaN
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if ~isnan(err)
    REFLPT_N_all(err(1,:),:)=NaN;
end

%normalization PNORM Vector
index_p=1:length(PNORM_N_all);
PNORM_N_all=PNORM_N_all(index_p,:)/repmat(sqrt(PNORM_N_all(index_p,1).^2 ...
    + PNORM_N_all(index_p,2).^2 + PNORM_N_all(index_p,3).^2),1,3);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear REFLPT_N_k DIREC_N_k NORMDI_N_k PNORM_N_k
clear REFLPT_N_k_1 DIREC_N_k_1 NORMDI_N_k_1 PNORM_N_k_1
clear REFLPT_N DIREC_N NORMDI_N PNORM_N
clear e1 e2 e3

%subfunction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function d = csort_h6(T)
%
%CSORT          T
%
%
%
%
if isempty(T)
    return
end

T=T(:)';

```

```

P=ones(size(T));
D = [T;P];
[Y,I] = sort(T);
X = D(:,I);
m = length(T);
TI = X(1,:);
PI = X(2,:);
j = 1;

t(1) = TI(1);
p(1) = PI(1);
for i = 1:(m - 1)
    if abs(TI(i) - TI(i+1)) < 1e-6 % 가 1e-6
        j = j; %
        p(j) = p(j) + PI(i+1);
    else
        j = j+1;
        t(j) = TI(i+1);

        p(j) = PI(i+1);
    end
end
d = [t;p];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

<<reflpoint_h6_new.m>>

```

function [REFLPT,DIREC,NORMDI,PNORM]=reflpoint_h6(imants,TXPOINT,RXPOINT)
% [REFLPT,DIREC,NORMDI,PNORM]=reflpoint_h6(imants,TXPOINT,RXPOINT)
%
% Return : REFLPT,DIREC,NORMDI,PNORM
% required function :

% File Log - SunHak Hong
% 11/01/99 - File Updated

% n
% : [REFLPT,DIREC]=reflpoint_h1(imants)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global ROOMSIZE

%Plane #
%x=0 ->1, y=0 ->2, z=0 ->3
%x=a ->4, y=b ->5, z=d ->6

TEMP=[eye(3);-eye(3)];

[lengthofimage,temp]=size(imants);

DIREC= repmat(RXPOINT,lengthofimage,1) - imants;

```

```

for index2=1:3
    alphap(:,index2)=(0- imants(:,index2)) / DIREC(:,index2);
    alphap(:,index2 +3 )=(ROOMSIZE(index2)- imants(:,index2)) / DIREC(:,index2);
end
clear index2

%distance=inf*ones(1,6);
distance=inf*ones(lengthofimage,6);

reflpoint_temp=[];
%reflpoint=[];
for index2=1:6
    reflpoint_temp = imants + DIREC .* repmat(alphap(:,index2),[1,3,1]);

    i=find( (reflpoint_temp(:,1) <=ROOMSIZE(1))&(reflpoint_temp(:,2) <=ROOMSIZE(2)) &...
        (reflpoint_temp(:,3) <=ROOMSIZE(3))&(reflpoint_temp(:,1) >= 0) &...
        (reflpoint_temp(:,2) >=0)&(reflpoint_temp(:,3) >=0) );

    distance(i,index2)=sqrt(sum((imants(i,:) - reflpoint_temp(i,:)).^2));
    clear reflpoint_temp
end
clear index2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[sort_distance,pre_loc]=sort(distance');
sort_distance=sort_distance';
pre_loc=pre_loc';

location=pre_loc(:,1);
i=find(sort_distance(:,1)==0);

location(i)=pre_loc(i,2);

% location(index 1)=find(~(distance- min(distance)));
alpha=zeros(lengthofimage,1);
for index3=1:lengthofimage
    alpha(index3,1)=alphap(index3,location(index3));
end
clear index3

REFLPT =imants + DIREC .* repmat(alpha,[1,3]);
NORMDI=TEMP(location,:);
PNORM=crossproduct_h6(DIREC,NORMDI);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function vect3=crossproduct_h6(vect1,vect2)

[lengthof,temp]=size(vect1);

vect3=[ vect1(:,2).*vect2(:,3)- vect1(:,3).*vect2(:,2),...

```

```

vect1(:,3).*vect2(:,1)-vect1(:,1).*vect2(:,3),...
vect1(:,1).*vect2(:,2)-vect1(:,2).*vect2(:,1)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

<<reflpointNk_h6_new.m>>

```

function
[REFLPT_N_k,DIREC_N_k,NORMDI_N_k,PNORM_N_k]=reflpointNk_h6(imants,TXPOINT,RXPOINT,REFLPT_N_k_1,
DIREC_N_k_1,NORMDI_N_k_1,refl_number)
%
[REFLPT_N_k,DIREC_N_k,NORMDI_N_k,PNORM_N_k]=reflpointNk_h6(imants,TXPOINT,RXPOINT,REFLPT_N_k_1,
DIREC_N_k_1,NORMDI_N_k_1,refl_number)
%
% Return : REFLPT_N_k,DIREC_N_k,NORMDI_N_k,PNORM_N_k
% required function :

% File Log - SunHak Hong
% 11/01/99 - File Updated

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global ROOMSIZE

```

```

%Plane #
% x=0 ->1, y=0 ->2, z=0 ->3
% x=a ->4, y=b ->5, z=d ->6

```

```

TEMP=[eye(3);-eye(3)];

```

```

[lengthof,temp]=size(REFLPT_N_k_1);

```

```

DIREC_N_k=(DIREC_N_k_1.*(abs(NORMDI_N_k_1)==0))+(DIREC_N_k_1.*(abs(NORMDI_N_k_1)==1).*(-1));

```

```

for index2=1:3
    alphap_N_k(:,index2)=(0-REFLPT_N_k_1(:,index2)) / DIREC_N_k(:,index2);
    alphap_N_k(:,index2 +3)=(ROOMSIZE(index2)-REFLPT_N_k_1(:,index2)) / DIREC_N_k(:,index2);
end

```

```

distance=inf*ones(lengthof,6);
reflpoint_N_k_temp=[];
reflpoint_N_k=[];

```

```

for index2=1:6
    reflpoint_N_k_temp=REFLPT_N_k_1 + DIREC_N_k .* repmat(alphap_N_k(:,index2),[1,3,1]);

    index1=find( (reflpoint_N_k_temp(:,1)<=ROOMSIZE(1))&(reflpoint_N_k_temp(:,2)<=ROOMSIZE(2))&...
        (reflpoint_N_k_temp(:,3)<=ROOMSIZE(3))&(reflpoint_N_k_temp(:,1)>=0)&...
        (reflpoint_N_k_temp(:,2)>=0)&(reflpoint_N_k_temp(:,3)>=0) );

    distance(index1,index2)=sqrt( sum( (REFLPT_N_k_1(index1,:) - reflpoint_N_k_temp(index1,:)).^2 ) );
    clear reflpoint_N_k_temp
end
clear index2 index1

```

```
[sort_distance,pre_loca]=sort(distance');
sort_distance=sort_distance';
pre_loca=pre_loca';

location=pre_loca(:,1);
i=find(sort_distance(:,1)==0);
location(i)=pre_loca(i,2);

alpha=zeros(lengthof,1);
for index3=1:lengthof
    alpha(index3,1)=alphap_N_k(index3,location(index3));
end
clear index3

REFLPT_N_k= REFLPT_N_k_1 + DIREC_N_k .* repmat(alpha,[1,3]);
NORMDI_N_k=TEMP(location,:);
PNORM_N_k=crossproduct_h6(DIREC_N_k,NORMDI_N_k);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function vect3=crossproduct_h6(vect1,vect2)

[lengthof,temp]=size(vect1);

vect3=[ vect1(:,2).*vect2(:,3)- vect1(:,3).*vect2(:,2),...
        vect1(:,3).*vect2(:,1)- vect1(:,1).*vect2(:,3),...
        vect1(:,1).*vect2(:,2)- vect1(:,2).*vect2(:,1)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

<<csort_h6.m>>

```
function d = csort_h6(T)
%
%CSORT          T
%
%              .
%              .
%              d=[          ;          ;          ]
%
% 1993.8.4      By Paul E. Pfeiffer
% 1998.2.7      modified by Lim Jong_Su
% 1999.9.6      modified by Hong SunHak

if isempty(T)
    d=[NaN;NaN;NaN];
    return
end

T=T(:)';
P=ones(size(T));
D = [T;P];
[Y,I] = sort(T);
X = D(:,I);
m = length(T);
TI = X(1,:);
```

```

PI = X(2,:);
j = 1;

t(1) = TI(1);
p(1) = PI(1);
for i = 1:(m - 1)
    if abs(TI(i) - TI(i+1)) < 1e-6 % 가 1e-6
        j = j; %
        p(j) = p(j) + PI(i+1);
    else
        j = j+1;
        t(j) = TI(i+1);

        p(j) = PI(i+1);
    end
end

location=I(cumsum(p));

d = [t;p;location];

```

<<path_angle_h7.m>>

```

function
[d_all,angle_incid]=path_anlge_h7(TXPOINT,RXPOINT,REFLPT_N_all,PNORM_N_all,NORMDI_N_all,imants,err)
% [d_all,angle_incid]=path_anlge_h7(TXPOINT,RXPOINT,REFLPT_N_all,PNORM_N_all,NORMDI_N_all,imants,err)
%
% Return : d_all,angle_incid
% required function :

% File Log - SunHak Hong
% 11/01/99 - File Updated

%path length & incident angle

```

```

global ROOMSIZE

```

```

global N

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
TEMP_B=triu(repmat(4*(1:N).^2 +2,N,1));
TEMP_C=cumsum(TEMP_B');
TEMP_D=TEMP_C(:,N);
TEMP_E=cumsum([0 ; TEMP_D(1:end-1)]);
TEMP_F=triu(TEMP_C+repmat(TEMP_E,1,N));
TEMP_G=triu(TEMP_F-TEMP_B+1);
index_start=triu(TEMP_G);
index_length=TEMP_B;
index_end=TEMP_F;

dN_vectors_all=[];
angle_incid_N=[];

```

```

d0_vectors_all=[];
angle_incid_0=[];
angle_incid_k=[];

for index=1:N
    %to RX

    index_1_N=index_start(1,index);
    index_2_N=index_end(1,index);
    dN_vectors= repmat(RXPOINT,[index_length(1,index),1])-REFLPT_N_all(index_1_N:index_2_N,:);
    numerator=sqrt( sum(( (dN_vectors).^2).*(~NORMDI_N_all(index_1_N:index_2_N,:)) )' ));
    denominator=sum(( abs(dN_vectors).*(NORMDI_N_all(index_1_N:index_2_N,:)) )' );
    dN_vectors_all=[dN_vectors_all ; dN_vectors];
    angle_of_incidence=(atan(sqrt(numerator)/(denominator)))';
    angle_incid_N=[angle_incid_N ; angle_of_incidence];
    clear numerator denominator
    clear dN_vectors angle_of_incidence

    %to TX
    index_1_0=index_start(index,index);
    index_2_0=index_end(index,index);
    d0_vectors= repmat(TXPOINT,[index_length(1,index),1])-REFLPT_N_all(index_1_0:index_2_0,:);
    numerator=sqrt( sum(( (d0_vectors).^2).*(~NORMDI_N_all(index_1_0:index_2_0,:)) )' ));
    denominator=sum(( abs(d0_vectors).*(NORMDI_N_all(index_1_0:index_2_0,:)) )' );
    d0_vectors_all=[d0_vectors_all ; d0_vectors];
    angle_of_incidence=(atan(sqrt(numerator)/(denominator)))';
    angle_incid_0=[angle_incid_0 ; angle_of_incidence];
    clear numerator denominator
    clear d0_vectors angle_of_incidence

    %REFLPT_k to REFLPT_k_1
    if index<N

        index_1_1=index_start(index,index+1);
        index_1_2=index_start(index+1,index+1);
        index_2_1=index_end(index,N);
        index_2_2=index_end(index+1,N);
        dk_vectors=( REFLPT_N_all(index_1_1:index_2_1,:)-REFLPT_N_all(index_1_2:index_2_2,:))^2 );
        numerator=sqrt( sum(( (dk_vectors).^2).*(~NORMDI_N_all(index_1_2:index_2_2,:)) )' ));
        denominator=sum(( abs(dk_vectors).*(NORMDI_N_all(index_1_2:index_2_2,:)) )' );
        angle_of_incidence=(atan(sqrt(numerator)/(denominator)))';
        dk_all(:,index)=zeros(index_end(1,N),1);
        dk_all(index_start(1,index+1):index_end(1,N),index)=sqrt( sum( dk_vectors' ) )';

        angle_incid_k=[angle_incid_k ; angle_of_incidence];
        clear numerator denominator
        clear dk_vectors angle_of_incidence

    end

end
end

```

```
dN_all=sqrt( sum( ( dN_vectors_all).^2 )' ); %NEW
clear dN_vectors
d0_all=sqrt( sum( ( d0_vectors_all).^2 )' ); %NEW
clear d0_vectors
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%summation of angle
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
angle_incid=[angle_incid_N ; angle_incid_k ; angle_incid_0];
clear angle_incid_N angle_incid_k angle_incid_0
d_all=sum ([dN_all' dk_all d0_all']');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

<<polari_h6.m>>

```
function
[pol_vectors,GAMMA_vectors,rho_s_total]=polari_h6(REFLPT_N_all,PNORM_N_all,NORMDI_N_all,pol_tx,angle_incid)
%
[pol_vectors,GAMMA_vectors,rho_s_total]=polari_h6(REFLPT_N_all,PNORM_N_all,NORMDI_N_all,pol_tx,angle_incid)
%
% Return : pol_vectors,GAMMA_vectors,rho_s_total
% required function :

% File Log - SunHak Hong
% 11/01/99 - File Updated

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
warning off
global ROOMSIZE
global N

[er_all,sigma_h,Pol_Compen_Factor]=er_h9(REFLPT_N_all);

global f
global lambda
global beta_a

pnorm=PNORM_N_all/ repmat( sqrt(sum( (PNORM_N_all.^2)'))',1,3);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
TEMP_B=triu(repmat(4*(1:N).^2 +2,N,1));
TEMP_C=cumsum(TEMP_B');
TEMP_D=TEMP_C(:,N);
TEMP_E=cumsum([0 ; TEMP_D(1:end- 1)]);
TEMP_F=triu(TEMP_C+repmat(TEMP_E,1,N));
TEMP_G=triu(TEMP_F- TEMP_B+1);
index_start=triu(TEMP_G);
index_length=TEMP_B;
index_end=TEMP_F;

pol_vectors=[];
rho_s_total=[];
```



```

GAMMA_vectors=[];

pol=pol_tx;
for index=1:N
    if index==1
        index_1=1;
    else
        index_1=index_end(index-1,N)+1;
    end

    index_2=index_end(index,N);

    if index==1
        pol_perpendi(index_1:index_2,:)=...
            repmat(sum((( repmat(pol,index_2-index_1+1,1)*(pnorm(index_1:index_2,:))'
).*eye(index_2))')',1,3).*pnorm(index_1:index_2,:);
        pol_parallel(index_1:index_2,:)=...
            repmat(pol,index_2-index_1+1,1)- pol_perpendi(index_1:index_2,:);
    else
        %nan      nan      nan
        pol_perpendi(index_1:index_2,:)=...
            repmat(sum((( pol*(pnorm(index_1:index_2,:))' ).*eye(length(pol)))')',1,3).*pnorm(index_1:index_2,:);
        pol_parallel(index_1:index_2,:)=...
            pol - pol_perpendi(index_1:index_2,:);
    end

    GAMMA_perpendi(index_1:index_2)=(cos(angle_incid(index_1:index_2)) - sqrt(er_all(index_1:index_2) -
sin(angle_incid(index_1:index_2)).^2)) / ...
        (cos(angle_incid(index_1:index_2)) + sqrt(er_all(index_1:index_2) - sin(angle_incid(index_1:index_2)).^2));
    GAMMA_parallel(index_1:index_2)=(er_all(index_1:index_2) .* cos(angle_incid(index_1:index_2)) -
sqrt(er_all(index_1:index_2)- sin(angle_incid(index_1:index_2)).^2)) / ...
        (er_all(index_1:index_2) .* cos(angle_incid(index_1:index_2)) + sqrt(er_all(index_1:index_2) -
sin(angle_incid(index_1:index_2)).^2));

    choo(index_1:index_2,:)=...
        [- (abs(NORMDI_N_all(index_1:index_2,1))==1) - (abs(NORMDI_N_all(index_1:index_2,2))==1)
        - (abs(NORMDI_N_all(index_1:index_2,3))==1)]+...
        [(abs(NORMDI_N_all(index_1:index_2,1))==0) (abs(NORMDI_N_all(index_1:index_2,2))==0)
        (abs(NORMDI_N_all(index_1:index_2,3))==0)];
    polv=( pol_perpendi(index_1:index_2,:).*repmat(GAMMA_perpendi(index_1:index_2),3,1)' +
    pol_parallel(index_1:index_2,:).
        .*repmat(GAMMA_parallel(index_1:index_2),3,1)' ).* choo(index_1:index_2,:);

    index_p=1:length(polv);
    GAMMA_pol=sqrt( abs(polv(:,1)).^2 +abs(polv(:,2)).^2 +abs(polv(:,3)).^2 );

    polv=polv(index_p,:)/ repmat(sqrt(polv(index_p,1).^2 + polv(index_p,2).^2 + polv(index_p,3).^2 +eps),1,3);

```

```

%scattering
rho_s=exp(-8*(pi*sigma_h(index_1:index_2,:).*sin((angle_incid(index_1:index_2,:))/lambda).^2) .*
besselj(0,-8*(pi*sigma_h(index_1:index_2,:).*sin(angle_incid(index_1:index_2,:))/lambda).^2);
rho_s_total=[rho_s_total rho_s];

pol_vectors=[pol_vectors ; polv];
GAMMA_vectors=[GAMMA_vectors ; GAMMA_pol];

%length(pol_vectors);
if index<N
    index_1_1=index_start(index,index+1);
    index_2_1=index_end(index,N);

    pol=pol_vectors(index_1_1:index_2_1,:);

    %normalize
    pol=pol./repmat(sqrt(pol(:,1).^2 + pol(:,2).^2 + pol(:,3).^2 +eps),1,3);

end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

<<er_h9.m>>

```

function [er_all,sigma_h,Pol_Compen_Factor]=er_h9(REFLPT_N_all)

%roughness
sigma_h=0.005*ones(length(REFLPT_N_all),1);
Pol_Compen_Factor=0.005*ones(length(REFLPT_N_all),1);

%ROOMSIZE=[6 8.15 3]
%concrete er=15
global f

e0=8.854*(10^-12);

wall_1 = 3 - j*5*(10^-4)/(2*pi*f*e0); %rainforced concrete, half height windows
wall_2 = 2.7 - j*5*(10^-3)/(2*pi*f*e0); %solid rainforced concrete
wall_3 = 3 - j*5*(10^-2)/(2*pi*f*e0); %plaster board and steel frame
wall_4 = 2.7 - j*5*(10^-3)/(2*pi*f*e0); %plaster board and timber frame
wall_5 = 1 - j*9*(10^6)/(2*pi*f*e0) ; %metal
wall_6 = 2 - j*3*(10^-4)/(2*pi*f*e0) ; %timber door

%initialization
%sqrt_er=sqrt(wall_2)*ones(length(REFLPT_N_all),1);
er_all = wall_2 * ones(length(REFLPT_N_all),1);

onwall_h9;

for index=1:length([onwallpatch])

    if strcmp(onwallpatch(index).material,'glass')
        erofwall=wall_1;
    elseif strcmp(onwallpatch(index).material,'metal')

```

```

erofwall=w all_5;
elseif strcmp(onwallpatch(index).material,'timber')
erofwall=w all_6;
else
erofwall=w all_1;
end

xcoor=[onwallpatch(index).coordi(1,1), onwallpatch(index).coordi(2,1)];
ycoor=[onwallpatch(index).coordi(1,2), onwallpatch(index).coordi(2,2)];
zcoor=[onwallpatch(index).coordi(1,3), onwallpatch(index).coordi(2,3)];

er_all( find(REFLPT_N_all(:,1)>=xcoor(1) & REFLPT_N_all(:,1)<=xcoor(2) ...
& REFLPT_N_all(:,2)>=ycoor(1) & REFLPT_N_all(:,2)<=ycoor(2) ...
& REFLPT_N_all(:,3)>=zcoor(1) & REFLPT_N_all(:,3)<=zcoor(2) ) )=erofwall;

sigma_h( find(REFLPT_N_all(:,1)>=xcoor(1) & REFLPT_N_all(:,1)<=xcoor(2) ...
& REFLPT_N_all(:,2)>=ycoor(1) & REFLPT_N_all(:,2)<=ycoor(2) ...
& REFLPT_N_all(:,3)>=zcoor(1) & REFLPT_N_all(:,3)<=zcoor(2) ) )=onwallpatch(index).roughness;
end

<<onwall2_h9.m>>
%2000/09/30

%y=0

y1=0;
glass_roughness=0;
metal_roughness=0;
timber_roughness=0;

onwallpatch(1).material='glass';
onwallpatch(1).roughness=glass_roughness;
onwallpatch(1).coordi=[1.85 y1 1.02 ;3.6 y1 2.88];

onwallpatch(2).material='glass';
onwallpatch(2).roughness=glass_roughness;
onwallpatch(2).coordi=[5.45 y1 1.02 ;7.2 y1 2.88];

onwallpatch(3).material='glass';
onwallpatch(3).roughness=glass_roughness;
onwallpatch(3).coordi=[9.05 y1 1.02 ;10.8 y1 2.88];

onwallpatch(4).material='glass';
onwallpatch(4).roughness=glass_roughness;
onwallpatch(4).coordi=[12.65 y1 1.02 ;14.4 y1 2.88];

onwallpatch(5).material='glass';
onwallpatch(5).roughness=glass_roughness;
onwallpatch(5).coordi=[18.05 y1 1.02 ;19.8 y1 2.88];

%y=xx
y2=8;

```

```

onw allpatch(6).material='glass';
onw allpatch(6).roughness=glass_roughness;
onw allpatch(6).coordi=[1.85 y2 1.02 ;3.6 y2 2.88];

onw allpatch(7).material='glass';
onw allpatch(7).roughness=glass_roughness;
onw allpatch(7).coordi=[5.45 y2 1.02 ;7.2 y2 2.88];

onw allpatch(8).material='glass';
onw allpatch(8).roughness=glass_roughness;
onw allpatch(8).coordi=[9.05 y2 1.02 ;10.8 y2 2.88];

onw allpatch(9).material='glass';
onw allpatch(9).roughness=glass_roughness;
onw allpatch(9).coordi=[12.65 y2 1.02 ;14.4 y2 2.88];

onw allpatch(10).material='glass';
onw allpatch(10).roughness=glass_roughness;
onw allpatch(10).coordi=[18.05 y2 1.02 ;19.8 y2 2.88];

```

```

%x=0
x1=0;

```

```

onw allpatch(11).material='glass';
onw allpatch(11).roughness=glass_roughness;
onw allpatch(11).coordi=[x1 1.35 1.02 ;x1 3.1 2.88];

```

```

onw allpatch(12).material='glass';
onw allpatch(12).roughness=glass_roughness;
onw allpatch(12).coordi=[x1 4.9 1.02 ;x1 6.65 2.88];

```

```

%x=yy
x2=21.65;

```

```

onw allpatch(13).material='glass';
onw allpatch(13).roughness=glass_roughness;
onw allpatch(13).coordi=[x2 1.72 1.1 ;x2 3.47 2.4];

```

```

onw allpatch(14).material='glass';
onw allpatch(14).roughness=glass_roughness;
onw allpatch(14).coordi=[x2 5.85 1.1 ;x2 7.6 2.4];

```

```

onw allpatch(15).material='metal';
onw allpatch(15).roughness=metal_roughness;
onw allpatch(15).coordi=[x2 1.72 0 ;x2 3.47 1.1];

```

```

onw allpatch(16).material='metal';
onw allpatch(16).roughness=metal_roughness;
onw allpatch(16).coordi=[x2 5.85 0 ;x2 7.6 1.1];

```

<<RCS_h9.m>>

```

function [EsOutScatter,TimeDelayScatter,PoolofCenter]=RCS_h9(TXPOINT,RXPOINT,pol_tx,tx_antenna_menu)
% [EsOutScatter,TimeDelayScatter]=RCS_h9(TXPOINT,RXPOINT,pol_tx,tx_antenna_menu)

```

```

global ROOMSIZE
global lambda

inwall_h9; %electrical properties of the desks
%%%header2

%%%pol_tx=(1/sqrt(2) 0 1/sqrt(2))/sqrt(sum([1/sqrt(2) 0 1/sqrt(2)])); %45Deg inclined
PoolofCenter=[];
for index=1:length(inwallpatch)

    if strcmp(inwallpatch(index).material,'timber')
        walltype=6;
    end

    sigma_h=inwallpatch(index).roughness;
    Center=inwallpatch(index).center;
    PoolofCenter=[PoolofCenter; Center];
    LxLy=inwallpatch(index).LxLy;
    InclinedAngle=inwallpatch(index).inclinedangle; % [x y z]degree

    Lx=LxLy(1);
    Ly=LxLy(2);
    v11=Center+[-Lx/2 -Ly/2 0];
    v22=Center+[ Lx/2  Ly/2 0];
    v12=Center+[ Lx/2 -Ly/2 0];
    v21=Center+[-Lx/2  Ly/2 0];
    p0=Center+[0 0 1]; %for the normal vector of the plane
    %normal_v=(p0-Center)/norm(p0-Center)

    Ver=[v11; v21; v22; v12];

    [TransVertices, normal_v]=affinetrans_h9(Center,Ver,InclinedAngle,p0); %affine transformation

    [EsOut,TimeDelay]=scatterpatch_h9(TXPOINT,RXPOINT,pol_tx,tx_antenna_menu,Center,LxLy,normal_v,sigma_h);

    EsOutScatter(index,:)=EsOut;
    TimeDelayScatter(index,:)=TimeDelay;
end

```

<<scatterpatch_h9.m>>

```

function
[EsOut,TimeDelay]=scatterpatch_h9(TXPOINT,RXPOINT,pol_tx,tx_antenna_menu,Center,LxLy,normal_v,sigma_h)
% [EsOut,TimeDelay]=scatterpatch_h9(TXPOINT,RXPOINT,pol_tx,tx_antenna_menu,Center,LxLy,normal_v)

```

```

global ROOMSIZE
global f
global lambda

```

```

%tx_antenna_menu=2;
%rx_antenna_menu=3;

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%inline funcions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%R_s , (alpha) : Fresnel reflection coefficient of patch for the polarization perpendicular to the plane of incidence
%R_p , (beta) : parallel to the plane of incidence
R_s_fun=inline('rho_s*(cos(theta_i)- sqrt(epsilon_r- (sin(theta_i)).^2))/(cos(theta_i)+sqrt(epsilon_r- (sin(theta_i)).^2))','rho_s','theta_i','epsilon_r');
R_p_fun=inline('rho_s*(epsilon_r*cos(theta_i)- sqrt(epsilon_r- (sin(theta_i)).^2))/(epsilon_r*cos(theta_i)+sqrt(epsilon_r- (sin(theta_i)).^2))','rho_s','theta_i','epsilon_r');
xi_x_fun=inline('sin(theta_i).*cos(varphi_i)- sin(theta_s).*cos(varphi_s)','theta_i','theta_s','varphi_i','varphi_s');
xi_y_fun=inline('sin(theta_i).*sin(varphi_i)- sin(theta_s).*sin(varphi_s)','theta_i','theta_s','varphi_i','varphi_s');
gamma_11_fun=inline('sin(theta_s)*sin(theta_i)*cos(theta_i)*(sin(varphi_s)*sin(varphi_i)+cos(varphi_s)*cos(varphi_i))*R_s','theta_i','theta_s','varphi_i','varphi_s','R_s');
gamma_12_fun=inline('sin(theta_s)*cos(theta_s)*sin(theta_i)*cos(theta_i)*(sin(varphi_s)*cos(varphi_i)- cos(varphi_s)*sin(varphi_i))*R_p','theta_i','theta_s','varphi_i','varphi_s','R_p');
gamma_21_fun=inline('sin(theta_s)*sin(theta_i)*(cos(varphi_s)*sin(varphi_i)- sin(varphi_s)*cos(varphi_i))*R_s','theta_i','theta_s','varphi_i','varphi_s','R_s');
gamma_22_fun=inline('sin(theta_s)*cos(theta_s)*sin(theta_i)*(cos(varphi_s)*cos(varphi_i)+sin(varphi_s)*sin(varphi_i))*R_p','theta_i','theta_s','varphi_i','varphi_s','R_p');
sigma_1k_fun=inline('(k^2 / pi *(L_x *L_y ) *( (sin(k*x_i_x*L_x/2))/(k* xi_x* L_x /2) ).^2 * (sin(k*xi_y*L_y/2))/(k* xi_y* L_y /2) ).^2 *abs(gamma_1k)^2',...
    'k','L_x','L_y','xi_x','xi_y','gamma_1k');
%d_Sj : distance between center of the jth patch and observation position
S_1k_fun=inline('sqrt(sigma_1k)/(sqrt(4*pi)*d_Sj)','sigma_1k','d_Sj');

rho_s_fun=inline('exp(- 8*((pi*sigma_h*sin(theta_i))/lambda).^2) .*
besseli(0,8*((pi*sigma_h*sin(theta_i))/lambda).^2)','sigma_h','theta_i','lambda');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%initial values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%f=1.89e9; lambda_0=3e9 / f; %global
lambda_0=lambda; %TEMP
k=2*pi/lambda_0;
e0=8.854*(10^- 12); %global
epsilon_r = 2 - j*3*(10^-4)/(2*pi*f*e0) ; %timber door
%epsilon_r = 1 - j*9*(10^6)/(2*pi*f*e0) ; %metal

%-----L_x=1.5;
%-----L_y=2.3;
L_x=LxLy(1);
L_y=Ly(2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%-----normv_patch=[0.1 0 1]/norm([0.1 0 1]);
normv_patch=normal_v;
%-----v_inci_ray=[.5 .5]/norm([.5 .5]); %

```

[illegible]

```

rho_s=rho_s_fun(sigma_h,pi-theta_i,lambda);

R_s=R_s_fun(rho_s,pi-theta_i,epsilon_r);
R_p=R_p_fun(rho_s,pi-theta_i,epsilon_r);
xi_x=xi_x_fun(theta_i,theta_s,varphi_i,varphi_s);
xi_y=xi_y_fun(theta_i,theta_s,varphi_i,varphi_s);

sigma_11=sigma_lk_fun(k,L_x,L_y,xi_x,xi_y,gamma_11_fun(theta_i,theta_s,varphi_i,varphi_s,R_s));
sigma_12=sigma_lk_fun(k,L_x,L_y,xi_x,xi_y,gamma_12_fun(theta_i,theta_s,varphi_i,varphi_s,R_p));
sigma_21=sigma_lk_fun(k,L_x,L_y,xi_x,xi_y,gamma_21_fun(theta_i,theta_s,varphi_i,varphi_s,R_s));
sigma_22=sigma_lk_fun(k,L_x,L_y,xi_x,xi_y,gamma_22_fun(theta_i,theta_s,varphi_i,varphi_s,R_p));

S11=S_lk_fun(sigma_11,d_Sj);
S12=S_lk_fun(sigma_12,d_Sj);
S21=S_lk_fun(sigma_21,d_Sj);
S22=S_lk_fun(sigma_22,d_Sj);

%Es_j : jth scattered field at the observation position
%Ei_j : jth incident field to the patch
%G_tj : radiation pattern of Tx antenna
%d_Tj : distance between Tx and center of the jth patch
Ei_j=E_0*G_tj*1/(4*pi*d_Tj) *exp(-j*k*d_Tj);

normv_inci_plane=cross(v_inci_ray,normv_patch);

%Ei_alpha1=sum(Ei_j .* normv_inci_plane) *normv_inci_plane
%Ei_beta1=normv_inci_plane - Ei_alpha1

%-----Ei_alpha1=sum(Ei_j .* normv_inci_plane) *normv_inci_plane
Ei_alpha1=sum(Ei_j .* normv_inci_plane);

v_temp=cross(normv_inci_plane,v_inci_ray); % (normv_inci_plane & v_inci_ray)
%-----Ei_beta1=sum(Ei_j .* v_temp) *v_temp
Ei_beta1=sum(Ei_j .* v_temp);

%incident and scattered ray-fixed coordinates represented by (d_1 alpha_1 beta_1), (d_2 alpha_2 beta_2)
%alpha : perpendicular
%beta : parallel
%for jth patch
Es=[S11 S12 ; S21 S22] * [Ei_alpha1 ; Ei_beta1]; % [Es_alpha2 ; Es_beta2]=[S11 S12 ; S21 S22] * [Ei_alpha1 ; Ei_beta1];
%Es_alpha2=Es(1,:)
%Es_beta2=Es(2,:)
scattv_temp=cross(v_scatt_ray,normv_inci_plane);

EsOut=Es(1,:)*normv_inci_plane+Es(2,:)*scattv_temp;

TimeDelay=(d_Tj+d_Sj)/(3*10^8);

<<inwall_h9.m>>

%

% (1-1)
inwallpatch(1).material='timber';
inwallpatch(1).roughness=0.001;

```



```

inwallpatch(1).center=[9.975      2.865      0.75];
inwallpatch(1).LxLy=[10.51      1.13]; %meter
inwallpatch(1).inclinedangle=[0      0      0]; % [x y z]degree

```

```

%      (1-2)
inwallpatch(2).material='timber';
inwallpatch(2).roughness=0.001;
inwallpatch(2).center=[9.975      5.125      0.75];
inwallpatch(2).LxLy=[10.51      1.13]; %meter
inwallpatch(2).inclinedangle=[0      0      0]; % [x y z]degree

```

```

%      (1-3)
inwallpatch(3).material='timber';
inwallpatch(3).roughness=0.001;
inwallpatch(3).center=[15.795      3.995      0.75];
inwallpatch(3).LxLy=[1.13      3.39]; %meter
inwallpatch(3).inclinedangle=[0      0      0]; % [x y z]degree

```

<<rx_antenna.m>>

```

function F_r=rx_antenna(theta,phi,thetaT_R,phiT_R,rx_antenna_menu)

```

```

if nargin==4
    rx_antenna_menu=3;
end

```

```

global theta_tilting
global phi_tilting
global lambda

```

```

theta=theta- thetaT_R+pi/ 2; %NEW
%theta=theta- theta_tilting;
phi=phi- phiT_R+(1e- 100)*(rx_antenna_menu>=4); %NEW;
%phi=phi- phi_tilting;

```

```

if rx_antenna_menu>=3
    omit=find((phi>pi/ 2)&(phi<3*pi/ 2));
    phi(omit)=nan;
end

```

```

switch rx_antenna_menu

```

```

case 1
    F_r=1;

```

```

case 2
    beampattern=sin(theta).^3;

```

```

    D=1.6866;
    %D=1;
    F_r=D*beampattern/ (max (max (beampattern)));

```

```

case 3
    % lby4 v
    kW=pi; kL=pi;
    E_phi=sin(theta) .* sin(kW/ 2 .* cos(theta)/cos(theta));

```

```

AF_slot_y=2*cos(kL/2 * sin(theta)).*sin(phi));

M=4; kdz=pi; psi_z=kdz * cos(theta);
S_zm=1/M * (sin(M/2 * psi_z)/sin(1/2 * psi_z));
beampattern=E_phi .*AF_slot_y .*S_zm;

aperture=0.35*0.05;
D=4*pi*aperture/(lambda^2);
%D=1;
F_r=D*beampattern/(max(max(beampattern)));

case 4
% lby4 h
kW=pi; kL=pi;
E_phi=sin(theta) .* sin(kW/2 .* cos(theta)/cos(theta));
AF_slot_y=2*cos(kL/2 * sin(theta)).*sin(phi));

N=4; kdy=pi; psi_y=kdy * sin(theta) .* sin(phi);
S_yn=1/N * (sin(N/2 * psi_y)/sin(1/2 * psi_y));
beampattern=E_phi .*AF_slot_y .*S_yn;

aperture=0.35*0.05;
D=4*pi*aperture/(lambda^2);
%D=1;
F_r=D*beampattern/(max(max(beampattern)));

case 5
% 4by4
kW=pi; kL=pi;
E_phi=sin(theta) .* sin(kW/2 .* cos(theta)/cos(theta));
AF_slot_y=2*cos(kL/2 * sin(theta)).*sin(phi));

N=4; kdy=pi; psi_y=kdy * sin(theta) .* sin(phi);
S_yn=1/N * (sin(N/2 * psi_y)/sin(1/2 * psi_y));

M=4; kdz=pi; psi_z=kdz * cos(theta);
S_zm=1/M * (sin(M/2 * psi_z)/sin(1/2 * psi_z));
beampattern=E_phi .*AF_slot_y .*S_yn .*S_zm;

aperture=0.35^2;
D=4*pi*aperture/(lambda^2);
%D=1;
F_r=D*beampattern/(max(max(beampattern)));
end

<<tx_antenna.m>>

function F_t=tx_antenna(theta,phi,thetaR_T,phiR_T,tx_antenna_menu)
%F_t=tx_antenna(theta,phi,thetaR_T,phiR_T,tx_antenna_menu)

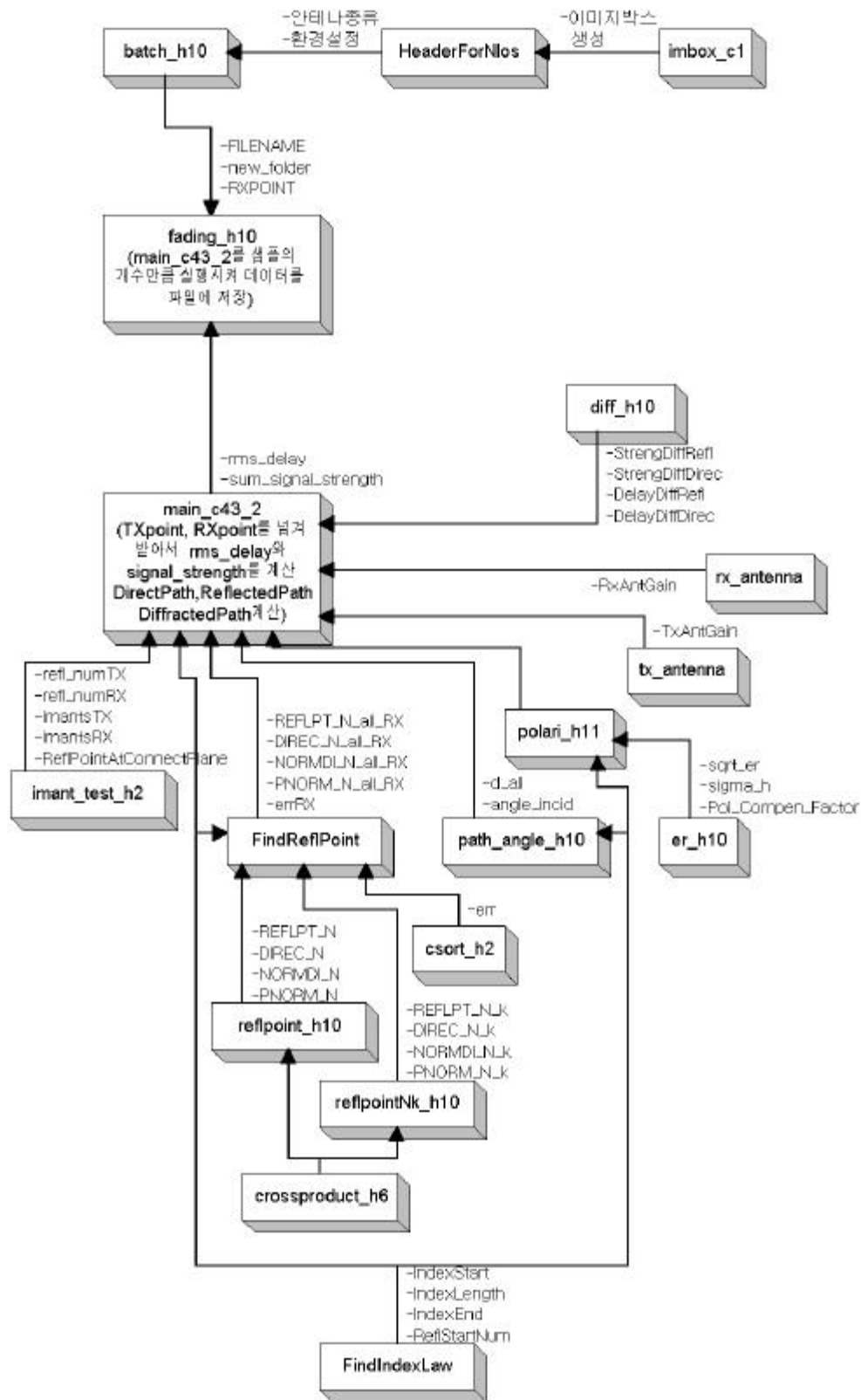
theta=theta-thetaR_T+pi/2; %NEW
phi=phi-phiR_T; %NEW

if tx_antenna_menu==1
F_t=1;

```

```
elseif tx_antenna_menu==2
    D=1.6866;
    %D=1;
    F_t=D*sin(theta);
end
```

Non Line of Sight MATLAB source code



block diagram

<<HeaderForNlos.m>>

function HeaderForNlos

```

global ROOMSIZERx; ROOMSIZERx=[7.97- 1.9 4.3 2.54];
%ROOMSIZERx=[30 7.2 3.5];
global ROOMSIZETx; ROOMSIZETx=[1.9 59.75 2.54] % [1.9 30 2.54];
%ROOMSIZETx=[8.4 34.8 3.5]; %offset + [13.2 7.2 0]
global OffsetRx; OffsetRx=[1.9 0 0];

global numofdiff_point; numofdiff_point=10;

global N; N=5;
global imboxs; imboxs=imbox_c1(N);
global f; f=1.895e9;
global lambda; lambda=3e8 /f;
global beta_a; beta_a=(2*pi)/lambda;
%global pol_tx; pol_tx=[0 0 1];

global pol_tx; pol_err=0.1; pol_tx=( [0 0 1]+pol_err)/sqrt(sum([0 0 1]+pol_err));
%pol_tx=( [1/sqrt(2) 0
1/sqrt(2)]+pol_err)/sqrt(sum([1/sqrt(2) 0 1/sqrt(2)]+pol_err)); %45Deg inclined

global POL_RX; POL_RX=[0 0 1 ; 0 1 0 ; 0 1/sqrt(2) 1/sqrt(2) ; 0 -1/sqrt(2) 1/sqrt(2)
];
global tx_antenna_menu; tx_antenna_menu=2; %Tx_ant dipole

global numofsample; numofsample=300;%numofsample=round(x_end-x_start)
%-----global TXPOINT_real; TXPOINT_real=[25.5 41.5 3.2];
global RXPOINT_pool; RXPOINT_pool=[4.65 0.5 2.3 ; 5.56 0.5 2.3];
%RXPOINT=[25.5 41.5 3.2];

%-----global RX_ROUTE;

global tx_height; tx_height=1.5;
global ROUTE
%-----x_start=1.1; % 14.5;
%-----x_end=1.1; % 25;
x_start=1.1; % 14.5;
x_end=1.1; % 25;

y_start=47.96; % 3.7;
y_end=12.96; % 0.5;

%-----
%
%Tx1
%---y_start=ROOMSIZETx(2)- 1
%---y_end=ROOMSIZETx(2)- 1
%Tx2
y_start=ROOMSIZETx(2)- 1- 15
y_end=ROOMSIZETx(2)- 1- 15
%Tx3

```

```

% --- y_start=ROOMSIZET x (2)- 1- 30
% --- y_end=ROOMSIZET x (2)- 1- 30
%Tx4
%y_start=ROOMSIZET x (2)- 1- 45
%y_end=ROOMSIZET x (2)- 1- 45
%-----

ROUTE(:,1)=[linspace(x_start, x_end +1e- 10, numofsample)' linspace(y_start, y_end +1e- 10, numofsample)'
tx_height*ones(1,numofsample)'];

```

<<imbox_c1.m>>

```

function imbox s=imbox_c1(N)

imbox s=[];
for index 1= -N:N
    for index 2= -N:N
        for index 3= -N:N

            if ( abs(index 1)+abs(index 2)+abs(index 3) <=N )&( abs(index 1)+abs(index 2)+abs(index 3) ~=0 )

                imbox s=[imbox s ; index 1 index 2 index 3];
            end

        end
    end
end

T=sum(abs(imbox s)')';
[sort_T,pre_loca]=sort(T);

imbox s=imbox s(pre_loca,:);

```

<<batch_h10.m>>

```

function batch_h10
% batch_h10
%
% Batch for fading_h10(FILENAMES,new_folder,RXPOINT)

% File Log - SunHak Hong
% 11/01/99 - File Updated

clear all

%global RX_ROUTE;
global RXPOINT_pool
HeaderForNlos;

%%%%%%%%%%%%%%
new_folder=input('Input the name of new data folder : ','s');
% --- new_folder='temp3';
status=mkdir(new_folder);

```

[illegible]

<<fading_h10.m>>

```
function [streng,rmsdelay]=fading_h10(FILENAMES,new_folder,RXPOINT)
% [streng,rmsdelay]=fading_h10(FILENAMES,new_folder,RXPOINT)
%
% Return : Signal Strength Vector & RMS Delay Spread Vector
% required function : main_c43_2

% File Log - SunHak Hong
% 11/01/99 - File Updated

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global ROOMSIZERx
global ROOMSIZETx
%%global OffsetRx

global numofdiff_point

global N
global imboxs
global f
global lambda
global beta_a

global pol_tx;                                %pol_tx=(1/sqrt(2) 0 1/sqrt(2)]+pol_err)/sqrt(sum([1/sqrt(2) 0
1/sqrt(2)]+pol_err)); %45Deg inclined

                                %pol_tx=(0 0 1]+pol_err)/sqrt(sum([0 0 1]+pol_err));
global POL_RX;                                %POL_RX=[0 0 1 ; 0 1 0 ; 0 1/sqrt(2) 1/sqrt(2) ; 0 -1/sqrt(2)
1/sqrt(2) ];

global tx_antenna_menu;                        %tx_antenna_menu=2; %Tx_ant dipole

global numofsample
%-----global RX_ROUTE
%-----global TXPOINT_real
global ROUTE
%-----global RXPOINT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%HeaderForNlos;

%-----TXPOINT=[29.5 41.5 3.2]-OffsetTx; %room2 %                                ->fit to room

[num_of_data,C_length]=size(FILENAMES);

STRENG=zeros(numofsample,num_of_data);
RMSDELAY=zeros(numofsample,num_of_data);

for index3=1:numofsample
    disp(FILENAMES(1,1:4));
    pp=sprintf('%d (%0.2g % % %)',index3,round(index3/numofsample*100000)*.001);
    disp(pp)
```



```

%-----RXPOINT=RX_ROUTE(index3,:);
TXPOINT=ROUTE(index3,:);

V_all=[]; delay_all=[];
[rms_delay,sum_signal_strength]=main_c43_2(TXPOINT,RXPOINT);
STRENG(index3,:)=20*log10(sum_signal_strength);
RMSDELAY(index3,:)=rms_delay;
end
filename_vectors=FILENAMES;
rmsdelay_vectors=RMSDELAY;
streng_vectors=STRENG;

XPD=mean(streng_vectors(:,1))-mean(streng_vectors(:,2))

cd(new_folder)

new_file=FILENAMES(1,1:4);
eval(['save ' new_file ' filename_vectors rmsdelay_vectors streng_vectors '])
cd('.')

figure
%x=RX_ROUTE(:,1,1);
plot((STRENG(:,1)),':');

ylabel('Signal Strength [dB]')
xlabel('Number of Sample')

%lcr_h7_spline(new_file,new_folder)
%lcr_h9(new_file,new_folder)
%correl_h9(new_file,new_folder)

<<main_c43_2.m>>

function [rms_delay,sum_signal_strength]=main_c43_2(TXPOINT,RXPOINT)
% [rms_delay,sum_signal_strength]=main_c43_2(TXPOINT,RXPOINT)
%
% Return : rms_delay,sum_signal_strength
% required function : imants_test_h2, FindIndexLaw, FindRefPoint, path_angle_h10, polari_h11,
% tx_antenna, rx_antenna, diff_h10

% File Log - SunHak Hong
% 11/01/99 - File Updated

global ROOMSIZETx
global ROOMSIZERx
global OffsetRx

global N
global imboxs

global f; %f=1.895e9;
global lambda; %lambda=3e8 /f;

```

```

global beta_a;      %beta_a=(2*pi)/lambda;
global pol_tx;      %pol_tx=[0 0 1];

%
%
%
imantsTX_temp=2*(-(imbox s<0)+(imbox s>=0)) .* repmat(ROOMSIZET x,length(imbox s),1) .*
fix(abs(imbox s+(imbox s>=0))/2) ...
+ repmat(TXPOINT ,length(imbox s),1) .* (-1).^(abs(imbox s));
%
%
%
RXPOINT_ForImage=RXPOINT - OffsetRx;
imantsRX_temp=2*(-(imbox s<0)+(imbox s>=0)) .* repmat(ROOMSIZERx,length(imbox s),1) .*
fix(abs(imbox s+(imbox s>=0))/2) ...
+ repmat(RXPOINT_ForImage,length(imbox s),1) .* (-1).^(abs(imbox s));
imantsRX_temp=imantsRX_temp+repmat(OffsetRx,length(imantsRX_temp),1); %offset
%
%
%
[refl_numTX,refl_numRX,imantsTX,imantsRX,RefIPointAtConnectPlane]=imants_test_h2(imantsTX_temp,imantsRX_tem
p);
[leng row]=size(imantsTX);
imantsRX_ForImage=imantsRX- repmat(OffsetRx,length(imantsRX),1);
%
%
Pre_refl_numTX=refl_numTX;
[refl_numTX,SortingIndexOfTx]=sort(refl_numTX);
imantsTX=imantsTX(SortingIndexOfTx,:);
OrigenOfIndexTX=[refl_numTX Pre_refl_numTX (1:length(SortingIndexOfTx))' SortingIndexOfTx ];
%
%
%
[IndexStartRx,IndexLengthRx,IndexEndRx,RefIStartNumRx]=FindIndexLaw (refl_numRX,N);

[IndexStartTx,IndexLengthTx,IndexEndTx,RefIStartNumTx]=FindIndexLaw (refl_numTX,N);
%
%
%RX
%
refl_num_r=refl_numRX;
imants_r=imantsRX_ForImage;
RoomSize_r=ROOMSIZERx;
RefIPointAtConnectPlane_ForImage=RefIPointAtConnectPlane- repmat(OffsetRx,leng,1);
[REFLPT_N_all_RX,DIREC_N_all_RX,NORMDI_N_all_RX,PNORM_N_all_RX,errRX]=FindRefIPoint (imants_r,refl_num_r
,RefIPointAtConnectPlane_ForImage,RoomSize_r);
%
REFLPT_N_all_RX=REFLPT_N_all_RX+repmat(OffsetRx,length(REFLPT_N_all_RX),1); %offset

%TX
%
refl_num_t=refl_numTX;

```

```
imants_t=imantsTX;
RoomSize_t=ROOMSIZETx;
[REFLPT_N_all_TX,DIREC_N_all_TX,NORMDI_N_all_TX,PNORM_N_all_TX,errTX]=FindRefPoint(imants_t,refl_num_
t,RefPointAtConnectPlane,RoomSize_t);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
errRX=errRX(1,:);
[d_all_RX,angle_incid_RX]=path_angle_h10(RXPOINT_ForImage,RefPointAtConnectPlane_ForImage,REFLPT_N_all_RX
,PNORM_N_all_RX,NORMDI_N_all_RX,imants_r,errRX,RoomSize_r,refl_num_r);

errTX=errTX(1,:);
[d_all_TX,angle_incid_TX]=path_angle_h10(TXPOINT_RefPointAtConnectPlane,REFLPT_N_all_TX,PNORM_N_all_TX,
NORMDI_N_all_TX,imants_t,errTX,RoomSize_t,refl_num_t);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%RX
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
REFLPT_N_all_RX=REFLPT_N_all_RX(end:- 1:1,:);
PNORM_N_all_RX=PNORM_N_all_RX(end:- 1:1,:);
NORMDI_N_all_RX=NORMDI_N_all_RX(end:- 1:1,:);
angle_incid_RX=angle_incid_RX(end:- 1:1,:);
[pol_vectors_RX,GAMMA_vectors_RX,rho_s_total_RX]=polari_h11(REFLPT_N_all_RX,PNORM_N_all_RX,NORMDI_N_a
ll_RX,angle_incid_RX,RoomSize_r,refl_num_r);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%TX
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
REFLPT_N_all_TX=REFLPT_N_all_TX(end:- 1:1,:);
PNORM_N_all_TX=PNORM_N_all_TX(end:- 1:1,:);
NORMDI_N_all_TX=NORMDI_N_all_TX(end:- 1:1,:);
angle_incid_TX=angle_incid_TX(end:- 1:1,:);
[pol_vectors_TX,GAMMA_vectors_TX,rho_s_total_TX]=polari_h11(REFLPT_N_all_TX,PNORM_N_all_TX,NORMDI_N_
all_TX,angle_incid_TX,RoomSize_t,refl_num_t); % offset %offset global

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
TxAntMenu=2; %
RX_ANT_MENU=[2 3 4 5]; %
POL_RX=[0 0 1 ; 0 1 0 ; 0 1/sqrt(2) 1/sqrt(2) ; 0 -1/sqrt(2) 1/sqrt(2) ]; %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%for Tx ant
DirectionTxAnt=[0 0 0]+TXPOINT.*[0 0 1]; %to fix of Tx antenna direction %[0 0 0]
dPT=DirectionTxAnt-TXPOINT;
thetaPT=acos(dPT(3)/sqrt(sum(dPT).^2));
phiPT=atan(dPT(2)/dPT(1));

%for Rx ant
DirectionRxAnt=ROOMSIZERx.*[1 1 0]+RXPOINT.*[0 0 1];
```

[illegible]

```

VrAll_tx=[];
for index=ReflStartNumTx:N
    index1=IndexStartTx(index,index);
    index2=IndexEndTx(index,index);
    reflpt1=REFLPT_N_all_TX(IndexStartTx(1,index):IndexEndTx(1,index),:);
    reflptN=REFLPT_N_all_TX(index1:index2,:);

    pol_N=pol_vectors_TX(index1:index2,:);

    Start=IndexStartTx(:,index);
    End=IndexEndTx(:,index);

    GammaN=ones(IndexLengthTx(1,index),1);
    Gamman=[];
    for indexG=1:index
        Gamman=GAMMA_vectors_TX(Start(indexG):End(indexG));
        GammaN=GammaN.*Gamman;
    end

    % scattering
    rho=rho_s_total_TX(index1:index2);

    diRT=reflpt1-repmat(TXPOINT,IndexLengthTx(index,index),1); %
    diTR=reflptN-ReflPointAtConnectPlane(IndexStartTx(1,index):IndexEndTx(1,index),:);

    rr=sqrt(sum((diRT).^2));

    if index==1
        PathLength=d_all_TX(1:IndexLengthTx(index,index))';
    else
        PathLength=d_all_TX(IndexStartTx(1,index):IndexEndTx(1,index))';
    end

    thetaiRT=acos(diRT(:,3)/rr);
    phiiRT=atan(diRT(:,2)/diRT(:,1));

    PathLoss_r=(1/PathLength).*exp(-j*beta_a*PathLength);
    delay_r=PathLength/(3e8);
    delay_r_all_tx=[delay_r_all_tx ; delay_r];

    TxAntGain_r=tx_antenna(thetaiRT,phiiRT,thetaPT,phiPT,TxAntMenu);
    Vr=[];
    for index1=2:(length(RX_ANT_MENU)+1)
        RxAntMenu=RX_ANT_MENU(index1-1);

        for index2=1:length(POL_RX) %polarization
            if ~(((RxAntMenu==3)|(RxAntMenu==4))&((index2==3)|(index2==4))) % omission of pol45 diversity at 1x4
%this line is also in batch_h9
                Vrtemp=TxAntGain_r .* PathLoss_r .* rho .*GammaN;
                Vr=[Vr Vrtemp];
            end
        end
    end
end

```

```

    VrAll_tx=[VrAll_tx ; Vr];
end

VrAll_tx=VrAll_tx (SortingIndexOfTx,:);
delay_r_all_tx=delay_r_all_tx (SortingIndexOfTx,:);

%reflected path : Rx to ReflPointAtConnectPlane
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
delay_r_all_rx=[];
VrAll_rx=[];
for index=ReflStartNumRx:N
    index1=IndexStartRx (index,index);
    index2=IndexEndRx (index,index);
    reflpt1=REFLPT_N_all_RX (IndexStartRx (1,index):IndexEndRx (1,index),:);
    reflptN=REFLPT_N_all_RX (index1:index2,:);

    pol_N=pol_vectors_RX (index1:index2,:);

    Start=IndexStartRx (:,index);
    Start=Start (find (Start~=0));
    End=IndexEndRx (:,index);
    End=End (find (End~=0));

    GammaN=ones (IndexLengthRx (1,index),1);
    Gamman=[];
    for indexG=1:index
        Gamman=GAMMA_vectors_RX (Start (indexG):End (indexG));
        GammaN=GammaN.*Gamman;
    end

    %scattering
    rho=rho_s_total_RX (index1:index2);

    diRT=reflpt1- ReflPointAtConnectPlane (IndexStartRx (1,index):IndexEndRx (1,index),:);
    diTR=reflptN- repmat (RXPOINT,IndexLengthRx (index,index),1);

    rr=sqrt (sum ( ( diRT).^2 )' )';

    if index==1
        PathLength=d_all_RX (1:IndexLengthRx (index,index))';
    else
        PathLength=d_all_RX (IndexStartRx (1,index) : IndexEndRx (1,index))';
    end

    thetaiTR=acos (diTR (:,3)/rr);
    phiiTR=- atan (diTR (:,2)/ diTR (:,1)); % -

    PathLoss_r=(1/PathLength).*exp (-j*beta_a*PathLength);
    delay_r=PathLength/ (3e8);
    delay_r_all_rx=[delay_r_all_rx ; delay_r];

    Vr=[];
    for index1=2:(length (RX_ANT_MENU)+1)

```

```

RxAntMenu=RX_ANT_MENU(index 1- 1);
RxAntGain_r=rx_antenna(thetaiT R,phiiT R,thetaPR,phiPR,RxAntMenu);

for index 2=1:length(POL_RX) %polarization
    if ~(((RxAntMenu==3)|(RxAntMenu==4))&((index 2==3)|(index 2==4))) %omission of pol45 diversity at 1x4
%this line is also in batch_h9
        Vrtemp=RxAntGain_r .* PathLoss_r .* sum( (pol_N .*
repmat(POL_RX(index 2,:),IndexLengthRx(index,index),1))' )' .*rho .*GammaN;
        Vr=[Vr Vrtemp];
    end
end
end
VrAll_rx=[VrAll_rx ; Vr];
end

delay_all=[LosDelay ; (delay_r_all_tx+delay_r_all_rx)];

V_all=[Vd ; (VrAll_tx+VrAll_rx)];%/IndexEndRx(1,N); %normalize

[StrengDiffRefl,StrengDiffDirec,DelayDiffRefl,DelayDiffDirec]=diff_h10(T XPOINT ,RXPOINT ,imantsRX_temp,imantsTX_
temp); %normalize

V_all=[V_all ; StrengDiffRefl ; StrengDiffDirec :];
delay_all=[delay_all ; DelayDiffRefl ; DelayDiffDirec];

[time_arrival,index_arrival]=sort(delay_all);
induced_voltage(1:length(index_arrival),:)=V_all(index_arrival(1:length(index_arrival),1),:);

[index_rms_2,kind_ant]=find( -20*log10(induced_voltage / repmat(max(induced_voltage),length(induced_voltage),1))
>= -50 );

[L_length,C_length]=size(induced_voltage);
rms_delay=[];
sum_signal_strength=[];
for index_column=1:C_length
    mean_delay=sum(abs(induced_voltage(index_rms_2,index_column)).*time_arrival(index_rms_2)) /
sum(abs(induced_voltage(index_rms_2,index_column)));
    second_moment=sum(abs(induced_voltage(index_rms_2,index_column)).*time_arrival(index_rms_2).^2) /
sum(abs(induced_voltage(index_rms_2,index_column)));
    rms_delay_temp=sqrt(second_moment - mean_delay.^2);
    rms_delay=[rms_delay rms_delay_temp];

    induced_voltage_temp=induced_voltage(:,index_column);
    IndexOfNaN=isnan(abs(induced_voltage_temp)); % 가
    induced_voltage_temp=induced_voltage_temp(~IndexOfNaN); % 가

    sum_signal_strength_temp=abs(sum(induced_voltage_temp(~isnan(induced_voltage_temp)).*exp(-i*2*pi*f*time_arrival(~
isnan(induced_voltage_temp)))));

    sum_signal_strength=[sum_signal_strength sum_signal_strength_temp];
end

induced_voltage=induced_voltage(:,1);
ImpulseResponse(time_arrival,induced_voltage,TXPOINT ,RXPOINT )

```

```

rms_delay*10^9
sum_signal_strength
pause

```

```

function ImpulseResponse(time_arrival,induced_voltage,TXPOINT,RXPOINT)

```

```

DirecDelay=(sqrt(abs(sum((TXPOINT - RXPOINT).^2)))/3e8);
h1=figure;
plot(time_arrival*(10^9),20*log10(abs(induced_voltage)/abs(max(induced_voltage))), 'linestyle','-');

```

```

set(gca,'xlim',[0 inf],'ylim',[- inf 1]);
xlabel('access delay (ns)');
ylabel('relative induced voltage [dB]');
title('Impluse Response');
set(gca,'xlim',[0 1200],'ylim',[- 70 0]);
%name_t1=sprintf('ROOMSIZE = [%1.2f %1.2f %1.2f]\nTXPOINT = [%1.2f %1.2f %1.2f]\nRXPOINT = [%1.2f %1.2f %1.2f]',ROOMSIZE,TXPOINT,RXPOINT);
%name_t2=sprintf('\nRX pol = [%1.2f %1.2f %1.2f]\nTX antenna = %s\nRX antenna = %s\nrms delay spread= %1.2f ns\nndirect path delay= %1.2f ns',pol_rx,tx_ant,rx_ant,rms_delay*10^9,DirecDelay*10^9);
%name_t3=sprintf('\nTheta tilting = %3.1fdeg\nPhi tilting = %3.1fdeg.',theta_tilting*180/pi,phi_tilting*180/pi);
%name_t=sprintf('Reflection Order (N) = %1.0f\nFrequency = %0.1g',N,f);
%set(h1,'numbertitle','off','name',name_t);
%tit=gtext([name_t1 name_t2]);
%set(tit,'HorizontalAlignment','left','VerticalAlignment','top');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

<<imant_test_h2.m>>

```

function
[refl_numTX,refl_numRX,imantsTX,imantsRX,RefPointAtConnectPlane]=imants_test_h2(imantsTX_temp,imantsRX_temp)
%
[refl_numTX,refl_numRX,imantsTX,imantsRX,RefPointAtConnectPlane]=imants_test_h2(imantsTX_temp,imantsRX_temp)
%
% Return : refl_numTX,refl_numRX,imantsTX,imantsRX,RefPointAtConnectPlane
% required function :

```

```

% File Log - SunHak Hong
% 11/01/99 - File Updated

```

```

global ROOMSIZETx
global ROOMSIZERx
global OffsetRx %for room2

```

```

global imboxs
global N

```

```

%find the maximum size of imantsRX matrix
TEMP_B=triu(repmat(4*(1:N).^2 +2,N,1));
TEMP_C=cumsum(TEMP_B');
TEMP_D=TEMP_C(:,N);
TEMP_E=cumsum([0 ; TEMP_D(1:end- 1)]);
TEMP_F=triu(TEMP_C+repmat(TEMP_E,1,N));

```



```

TEMP_G=triu(TEMP_F-TEMP_B+1);
index_start=triu(TEMP_G);
index_length=TEMP_B;
index_end=TEMP_F;

%imants direc vectors
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x.direc_temp= repmat(imantsRX_temp(:,1),[1,index_end(1,N)])'- repmat(imantsTX_temp(:,1),[1,index_end(1,N)]);%
    7}
y.direc_temp= repmat(imantsRX_temp(:,2),[1,index_end(1,N)])'- repmat(imantsTX_temp(:,2),[1,index_end(1,N)]);
z.direc_temp= repmat(imantsRX_temp(:,3),[1,index_end(1,N)])'- repmat(imantsTX_temp(:,3),[1,index_end(1,N)]);

direc_temp2(:,1)= reshape(x.direc_temp,[index_end(1,N)^2,1]);
direc_temp2(:,2)= reshape(y.direc_temp,[index_end(1,N)^2,1]);
direc_temp2(:,3)= reshape(z.direc_temp,[index_end(1,N)^2,1]);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%room1    room2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%----- alpha=( OffsetRx(2) - repmat(imantsTX_temp(:,2),[index_end(1,N),1]) )/ direc_temp2(:,2);%RX7}
:      RX      TX
alpha=( OffsetRx(1) - repmat(imantsTX_temp(:,1),[index_end(1,N),1]) )/ direc_temp2(:,1);
reflect= repmat(imantsTX_temp,[index_end(1,N),1]) + direc_temp2 .* repmat(alpha,[1,3]);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%reflection plane condition
result_vector_temp= find( (reflect(:,2) >=0)&(reflect(:,2) <= (ROOMSIZERx(2)))&(reflect(:,3) >=0) & (reflect(:,3)
<=ROOMSIZERx(3)) );

imantsTX_bank= repmat(imantsTX_temp,[index_end(1,N),1]);
imantsTX= imantsTX_bank(result_vector_temp,:);

imantTX_index= rem(result_vector_temp,index_end(1,N));
imantRX_index= fix(result_vector_temp/(index_end(1,N)+0.01))+1;

imantsRX= imantsRX_temp(imantRX_index,:);

%
ReflPointAtConnectPlane= reflect(result_vector_temp,:);

%reflection number of Tx and Rx
refl_numTX= zeros(size(imantTX_index));
refl_numRX= zeros(size(imantRX_index));
for index_refl=N:-1:1
    TEMP1= imantTX_index - index_end(1,index_refl);
    refl_numTX( find(TEMP1<=0) )= index_refl;
    TEMP2= imantRX_index - index_end(1,index_refl);
    refl_numRX( find(TEMP2<=0) )= index_refl;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

<<FindIndexLaw.m>>

function [IndexStart,IndexLength,IndexEnd,ReflStartNum]=FindIndexLaw(refl_num,N)

% [IndexStart,IndexLength,IndexEnd,ReflStartNum]=FindIndexLaw(refl_num,N)

```
%
% Return : IndexStart,IndexLength,IndexEnd,ReflStartNum
% required function :
```

```
% File Log - SunHak Hong
% 11/01/99 - File Updated
```

```
Y=csort_h2(refl_num);
ReflStartNum=Y(1,1);
```

```
[LengthY, ColumY]=size(Y);
```

```
if ColumY~=N
```

```
    for index=Y(1,1)-1:-1:1
        InsertY=[1 0 0]';
        Y=[InsertY Y];
    end
```

```
end
```

```
[LengthY, ColumY]=size(Y);
```

```
if ColumY~=1
```

```
    YY=zeros(3,N);
    for index=1:ColumY
        YY(:,Y(1,index))=Y(:,index);
    end
```

```
    Y=YY;
```

```
end
```

```
TEMP_B=triu(repmat(Y(2,:),N,1));
TEMP_C=cumsum(TEMP_B')';
TEMP_D=TEMP_C(:,N);
TEMP_E=cumsum([0 ; TEMP_D(1:end-1)]);
TEMP_F=triu(TEMP_C+repmat(TEMP_E,1,N));
TEMP_G=triu(TEMP_F-TEMP_B+1);
IndexStart=triu(TEMP_G);
IndexLength=TEMP_B;
IndexEnd=TEMP_F;
```

<<FindReflPoint.m>>

```
function [REFLPT,DIREC,NORMDI,PNORM]=reflpoint_h10(imants,ReflPointAtConnectPlane,ROOMSIZE)
```

```
%Plane #
```

```
%x=0 ->1, y=0 ->2, z=0 ->3
```

```
%x=a ->4, y=b ->5, z=d ->6
```

```
TEMP=[eye(3):-eye(3)];
```

```
[lengthofimage,temp]=size(imants);
```

```
%DIREC=repmat(RXPOINT,lengthofimage,1) - imants;
```

```
DIREC=ReflPointAtConnectPlane-imants;
```

```

for index2=1:3
    alphap(:,index2)=(0-imants(:,index2)) / DIREC(:,index2);
    alphap(:,index2 +3 )=(ROOMSIZE(index2)-imants(:,index2)) / DIREC(:,index2);
end
clear index2

%distance=inf*ones(1,6);
distance=inf*ones(lengthofimage,6);

reflpoint_temp=[];
%reflpoint=[];
for index2=1:6
    reflpoint_temp = imants + DIREC .* repmat(alphap(:,index2),[1,3,1]);

    i=find( (reflpoint_temp(:,1) <=ROOMSIZE(1))&(reflpoint_temp(:,2) <=ROOMSIZE(2)) &...
        (reflpoint_temp(:,3) <=ROOMSIZE(3))&(reflpoint_temp(:,1) >= 0) &...
        (reflpoint_temp(:,2) >=0)&(reflpoint_temp(:,3) >=0) );

    distance(i,index2)=sqrt(sum((imants(i,:) - reflpoint_temp(i,:)).^2));
    clear reflpoint_temp
end
clear index2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[sort_distance,pre_loca]=sort(distance');
sort_distance=sort_distance';
pre_loca=pre_loca';

location=pre_loca(:,1);
i=find(sort_distance(:,1)==0);

location(i)=pre_loca(i,2);

%location(index1)=find(~(distance-min(distance)));
alpha=zeros(lengthofimage,1);
for index3=1:lengthofimage
    alpha(index3,1)=alphap(index3,location(index3));
end
clear index3

REFLPT=imants + DIREC .* repmat(alpha,[1,3]);
NORMDI=TEMP(location,:);
PNORM=crossproduct_h6(DIREC,NORMDI);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function vect3=crossproduct_h6(vect1,vect2)

[lengthof,temp]=size(vect1);

vect3=[ vect1(:,2).*vect2(:,3)-vect1(:,3).*vect2(:,2),...
        vect1(:,3).*vect2(:,1)-vect1(:,1).*vect2(:,3),...

```

%%%%%%%%%

```
for index3=1:lengthof
```

```

        alpha(index3,1)=alphap_N_k(index3,location(index3));
end
clear index3

REFLPT_N_k= REFLPT_N_k_1 + DIREC_N_k .* repmat(alpha,[1,3]);
NORMDI_N_k=TEMP(location,:);
PNORM_N_k=crossproduct_h6(DIREC_N_k,NORMDI_N_k);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function vect3=crossproduct_h6(vect1,vect2)

[lengthof,temp]=size(vect1);

vect3=[ vect1(:,2).*vect2(:,3)-vect1(:,3).*vect2(:,2),...
        vect1(:,3).*vect2(:,1)-vect1(:,1).*vect2(:,3),...
        vect1(:,1).*vect2(:,2)-vect1(:,2).*vect2(:,1)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

<<csort_h2.m>>

```

function d = csort_h2(T)
%
% CSORT          T
%
%
%
%          d=[          ;          ;          ]
%
% 1993.8.4      By Paul E. Pfeiffer
% 1998.2.7      modified by Lim Jong_Su
% 1999.9.6      modified by Hong SunHak

if isempty(T)
    d=[NaN;NaN;NaN];
    return
end

T=T(:)';
P=ones(size(T));
D = [T;P];
[Y,I] = sort(T);
X = D(:,I);
m = length(T);
TI = X(1,:);
PI = X(2,:);
j = 1;

t(1) = TI(1);
p(1) = PI(1);
for i = 1:(m - 1)
    if abs(TI(i) - TI(i+1)) < 1e-6 %          가 1e-6
        j = j; %          .
        p(j) = p(j) + PI(i+1);
    end
end

```

```

else
    j = j+1;
    t(j) = TI(i+1);

    p(j) = PI(i+1);
end
end

```

```
location=I(cumsum(p));
```

```
d = [t;p;location];
```

<<path_angle_h 10.m>>

```

function
[d_all,angle_incid]=path_angle_h10(TXPOINT,RXPOINT,REFLPT_N_all,PNORM_N_all,NORMDI_N_all,imants,err,ROOMSIZE,refl_num)
%
[d_all,angle_incid]=path_angle_h10(TXPOINT,RXPOINT,REFLPT_N_all,PNORM_N_all,NORMDI_N_all,imants,err,ROOMSIZE,refl_num)
%
% Return : d_all,angle_incid
% required function : FindIndexLaw

% File Log - SunHak Hong
% 11/01/99 - File Updated

%pass length & incident angle

warning off

%global ROOMSIZE
global N

%direct_pass
[IndexStart,IndexLength,IndexEnd,ReflStartNum]=FindIndexLaw(refl_num,N);
index_start=IndexStart;
index_length=IndexLength;
index_end=IndexEnd;

dN_vectors_all=[];
angle_incid_N=[];
d0_vectors_all=[];
angle_incid_0=[];
angle_incid_k=[];

for index=1:N
    %to RX

    if index>=ReflStartNum
        index_1_N=index_start(1,index);
        index_2_N=index_end(1,index);

```

```

% ---- dN_vectors= repmat(RXPOINT,[index_length(1,index),1])- REFLPT_N_all(index_1_N:index_2_N,:);
dN_vectors=RXPOINT(index_1_N:index_2_N,:)- REFLPT_N_all(index_1_N:index_2_N,:);
numerator=sqrt( sum( ( (dN_vectors).^2).*(~NORMDI_N_all(index_1_N:index_2_N,:)) )' );
denominator=sum( ( abs(dN_vectors).*(NORMDI_N_all(index_1_N:index_2_N,:)) )' );
dN_vectors_all=[dN_vectors_all ; dN_vectors];
angle_of_incidence=(atan(sqrt(numerator)/(denominator)))';
angle_incid_N=[angle_incid_N ; angle_of_incidence];
clear numerator denominator
clear dN_vectors angle_of_incidence

%to TX
index_1_0=index_start(index,index);
index_2_0=index_end(index,index);
d0_vectors= repmat(TXPOINT,[index_length(1,index),1])- REFLPT_N_all(index_1_0:index_2_0,:);
numerator=sqrt( sum( ( (d0_vectors).^2).*(~NORMDI_N_all(index_1_0:index_2_0,:)) )' );
denominator=sum( ( abs(d0_vectors).*(NORMDI_N_all(index_1_0:index_2_0,:)) )' );
d0_vectors_all=[d0_vectors_all ; d0_vectors];
angle_of_incidence=(atan(sqrt(numerator)/(denominator)))';
angle_incid_0=[angle_incid_0 ; angle_of_incidence];
clear numerator denominator
clear d0_vectors angle_of_incidence

end

%REFLPT_k to REFLPT_k_1
if index<N
    index_1_1=index_start(index,index+1);
    index_1_2=index_start(index+1,index+1);
    index_2_1=index_end(index,N);
    index_2_2=index_end(index+1,N);
    dk_vectors=( REFLPT_N_all(index_1_1:index_2_1,:)- REFLPT_N_all(index_1_2:index_2_2,:)).^2 );
    numerator=sqrt( sum( ( (dk_vectors).^2).*(~NORMDI_N_all(index_1_2:index_2_2,:)) )' );
    denominator=sum( ( abs(dk_vectors).*(NORMDI_N_all(index_1_2:index_2_2,:)) )' );
    angle_of_incidence=(atan(sqrt(numerator)/(denominator)))';
    dk_all(:,index)=zeros(index_end(1,N),1);
    dk_all(index_start(1,index+1):index_end(1,N),index)=sqrt( sum( dk_vectors' ) );

    angle_incid_k=[angle_incid_k ; angle_of_incidence];
    clear numerator denominator
    clear dk_vectors angle_of_incidence
end

end

dN_all=sqrt( sum( ( (dN_vectors_all).^2 )' ) ); %NEW
clear dN_vectors
d0_all=sqrt( sum( ( (d0_vectors_all).^2 )' ) ); %NEW
clear d0_vectors

%summation of angle
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
angle_incid=[angle_incid_N ; angle_incid_k ; angle_incid_0];
clear angle_incid_N angle_incid_k angle_incid_0

```

```
d_all=sum([dN_all' dk_all d0_all']');
d_all';
```

<<polari_h11.m>>

```
function
[pol_vectors,GAMMA_vectors,rho_s_total]=polari_h10(REFLPT_N_all,PNORM_N_all,NORMDI_N_all,angle_incid,ROOM
SIZE,refl_num)
%
[pol_vectors,GAMMA_vectors,rho_s_total]=polari_h10(REFLPT_N_all,PNORM_N_all,NORMDI_N_all,angle_incid,ROOM
SIZE,refl_num)
%
% Return : pol_vectors,GAMMA_vectors,rho_s_total
% required function : er_h10, FindIndexLaw

% File Log - SunHak Hong
% 11/01/99 - File Updated

warning off
%global ROOMSIZE
global N

%er_h6
%[sqrt_er,sigma_h,Pol_Compen_Factor]=er_h7_2(REFLPT_N_all);
%edit new version 99/08/21
[er_all,sigma_h,Pol_Compen_Factor]=er_h10(REFLPT_N_all);

global f
global lambda
global beta_a
global pol_tx

pnorm=PNORM_N_all/ repmat( sqrt(sum( (PNORM_N_all.^2)'))',1,3);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[IndexStart,IndexLength,IndexEnd,ReflStartNum]=FindIndexLaw(refl_num,N);
index_start=IndexStart;
index_length=IndexLength;
index_end=IndexEnd;

pol_vectors=[];
rho_s_total=[];
GAMMA_vectors=[];

pol=pol_tx;
for index=ReflStartNum:N
    if index==ReflStartNum
        index_1=1;
    else
```



```

        index_1=index_end(index - 1,N)+1;
    end

    index_2=index_end(index,N);
    [leng_pol,temp]=size(pol);

    if index==ReflStartNum
        pol_perpendi(index_1:index_2,:)=...
            repmat(sum((( repmat(pol,index_2- index_1+1,1)*(pnorm(index_1:index_2,:))' )
).*eye(index_2))',1,3).*pnorm(index_1:index_2,:);
        pol_parallel(index_1:index_2,:)=...
            repmat(pol,index_2- index_1+1,1)- pol_perpendi(index_1:index_2,:);
    else
        %nan        nan    nan

        pol_perpendi(index_1:index_2,:)=...
            repmat(sum((( pol*(pnorm(index_1:index_2,:))' ).*eye(leng_pol))',1,3).*pnorm(index_1:index_2,:);
        pol_parallel(index_1:index_2,:)=...
            pol - pol_perpendi(index_1:index_2,:);
    end

    %keyboard
    %new - version
    GAMMA_perpendi(index_1:index_2)=(cos(angle_incid(index_1:index_2)) - sqrt(er_all(index_1:index_2) -
sin(angle_incid(index_1:index_2)).^2)) / ...
        (cos(angle_incid(index_1:index_2)) + sqrt(er_all(index_1:index_2) - sin(angle_incid(index_1:index_2)).^2));
    GAMMA_parallel(index_1:index_2)=(er_all(index_1:index_2) .* cos(angle_incid(index_1:index_2)) -
sqrt(er_all(index_1:index_2)- sin(angle_incid(index_1:index_2)).^2)) / ...
        (er_all(index_1:index_2) .* cos(angle_incid(index_1:index_2)) + sqrt(er_all(index_1:index_2) -
sin(angle_incid(index_1:index_2)).^2));

    choo(index_1:index_2,:)=...
        [-(abs(NORMDI_N_all(index_1:index_2,1))==1) - (abs(NORMDI_N_all(index_1:index_2,2))==1)
- (abs(NORMDI_N_all(index_1:index_2,3))==1)]+...
        [(abs(NORMDI_N_all(index_1:index_2,1))==0) (abs(NORMDI_N_all(index_1:index_2,2))==0)
(abs(NORMDI_N_all(index_1:index_2,3))==0)];
    polv=( pol_perpendi(index_1:index_2,:).*repmat(GAMMA_perpendi(index_1:index_2),3,1)' +
pol_parallel(index_1:index_2,:)...
        .*repmat(GAMMA_parallel(index_1:index_2),3,1))' .* choo(index_1:index_2,:);

    index_p=1:leng_pol;
    GAMMA_pol=sqrt( abs(polv(:,1)).^2 +abs(polv(:,2)).^2 +abs(polv(:,3)).^2 );

    polv=polv(index_p,:)/ repmat(sqrt(polv(index_p,1).^2 + polv(index_p,2).^2 + polv(index_p,3).^2 +eps),1,3);

    %scattering
    rho_s=exp(- 8*(pi*sigma_h(index_1:index_2,:).*sin((angle_incid(index_1:index_2,:))/ lambda).^2) .*

```

```

besselj(0,-8*(pi*sigma_h(index_1:index_2,:).*sin(angle_incid(index_1:index_2,:))/lambda).^2);
rho_s_total=[rho_s_total rho_s];

pol_vectors=[pol_vectors ; polv];
GAMMA_vectors=[GAMMA_vectors ; GAMMA_pol];

index
size(pol_vectors)
%length(pol_vectors);
if index<N
    index_1_1=index_start(index,index+1);
    index_2_1=index_end(index,N);

    pol=pol_vectors(index_1_1:index_2_1,:);

    [leng_pol,temp]=size(pol);

    %normalize
    pol=pol./repmat(sqrt(pol(:,1).^2 + pol(:,2).^2 + pol(:,3).^2 +eps),1,3);
end

end

%%%%%%%%%%

```

<<er_h10.m>>

```

function [er_all,sigma_h,Pol_Compen_Factor]=er_h10(REFLPT_N_all)

%roughness
sigma_h=0.005*ones(length(REFLPT_N_all),1);
Pol_Compen_Factor=0.005*ones(length(REFLPT_N_all),1);

%ROOMSIZE=[6 8.15 3]
%concrete er=15
global f

e0=8.854*(10^-12);

wall_1 = 3 - j*5*(10^-4)/(2*pi*f*e0); %rainforced concrete, half height windows
wall_2 = 2.7 - j*5*(10^-3)/(2*pi*f*e0); %solid rainforced concrete
wall_3 = 3 - j*5*(10^-2)/(2*pi*f*e0); %plaster board and steel frame
wall_4 = 2.7 - j*5*(10^-3)/(2*pi*f*e0); %plaster board and timber frame
wall_5 = 1 - j*9*(10^6)/(2*pi*f*e0) ; %metal
wall_6 = 2 - j*3*(10^-4)/(2*pi*f*e0) ; %timber door

%initialization
%sqrt_er=sqrt(wall_2)*ones(length(REFLPT_N_all),1);
er_all = wall_2 * ones(length(REFLPT_N_all),1);

onwall2_h9;

```

```

for index=1:length([onwallpatch])

    if strcmp(onwallpatch(index).material,'glass')
        erofwall=wall_1;
    elseif strcmp(onwallpatch(index).material,'metal')
        erofwall=wall_5;
    elseif strcmp(onwallpatch(index).material,'timber')
        erofwall=wall_6;
    else
        erofwall=wall_1;
    end

    xcoor=[onwallpatch(index).coordi(1,1), onwallpatch(index).coordi(2,1)];
    ycoor=[onwallpatch(index).coordi(1,2), onwallpatch(index).coordi(2,2)];
    zcoor=[onwallpatch(index).coordi(1,3), onwallpatch(index).coordi(2,3)];

    er_all( find(REFLPT_N_all(:,1)>=xcoor(1) & REFLPT_N_all(:,1)<=xcoor(2) ...
        & REFLPT_N_all(:,2)>=ycoor(1) & REFLPT_N_all(:,2)<=ycoor(2) ...
        & REFLPT_N_all(:,3)>=zcoor(1) & REFLPT_N_all(:,3)<=zcoor(2) ) )=erofwall;

    sigma_h( find(REFLPT_N_all(:,1)>=xcoor(1) & REFLPT_N_all(:,1)<=xcoor(2) ...
        & REFLPT_N_all(:,2)>=ycoor(1) & REFLPT_N_all(:,2)<=ycoor(2) ...
        & REFLPT_N_all(:,3)>=zcoor(1) & REFLPT_N_all(:,3)<=zcoor(2) ) )=onwallpatch(index).roughness;
end

<<diff_h 10.m>>

function
[StrengDiffRefl,StrengDiffDirec,DelayDiffRefl,DelayDiffDirec]=diff_h 10(T XPOINT ,RXPOINT ,imantsRX_temp,imantsTX_
temp)
%
[StrengDiffRefl,StrengDiffDirec,DelayDiffRefl,DelayDiffDirec]=diff_h 10(T XPOINT ,RXPOINT ,imantsRX_temp,imantsTX_
temp)
%
% Return : StrengDiffRefl,StrengDiffDirec,DelayDiffRefl,DelayDiffDirec
% required function :

% File Log - SunHak Hong
% 11/01/99 - File Updated

global ROOMSIZETx
global ROOMSIZERx
global OffsetRx %for room2

global imboxs
global N

global numofdiff_point

```



```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
los_length1=sqrt(sum((abs(diff_point-repmat(RXPOINT,numofdiff_point,1)).^2),2)); %Rx
los_delay1=los_length1/(3e8);
los_loss1=(1/los_length1).*exp(-j*beta_a*los_delay1);

streng_at_d_los=los_loss1/((numofdiff_point*(index_end(1,N)+1)));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% beta_0 pi_angle_p pi_angle
x_unit=repmat([1 0 0],numofdiff_point*index_end(1,N),1);
y_unit=repmat([0 1 0],numofdiff_point*index_end(1,N),1);
z_unit=repmat([0 0 1],numofdiff_point*index_end(1,N),1);
beta_0=acos(sum(dot(direct_d2TX,z_unit,length(z_unit)),2)/sqrt(sum(direct_d2TX.^2,2)).*1));
beta_0_p=acos(sum(dot(direct_RX2d,z_unit,length(z_unit)),2)/sqrt(sum(direct_RX2d.^2,2)).*1));
pi_angle_p=acos(sum(dot(direct_RX2d,y_unit,length(y_unit)),2)/sqrt(sum(direct_RX2d.^2,2)).*1));
pi_angle=(3*pi/2)-acos(sum(dot(direct_d2TX,x_unit,length(x_unit)),2)/sqrt(sum(direct_d2TX.^2,2)).*1));

% beta_0_los pi_angle_p_los pi_angle_los

x_unit_l=repmat([1 0 0],numofdiff_point,1);
y_unit_l=repmat([0 1 0],numofdiff_point,1);
z_unit_l=repmat([0 0 1],numofdiff_point,1);
beta_0_los=acos(sum(dot(direct_dtoTX,z_unit_l,length(z_unit_l)),2)/sqrt(sum(direct_dtoTX.^2,2)).*1));
beta_0_los_p=acos(sum(dot(direct_RXtod,z_unit_l,length(z_unit_l)),2)/sqrt(sum(direct_RXtod.^2,2)).*1));
pi_angle_p_los=acos(sum(dot(direct_RXtod,y_unit_l,length(y_unit_l)),2)/sqrt(sum(direct_RXtod.^2,2)).*1));
pi_angle_los=(3*pi/2)-acos(sum(dot(direct_dtoTX,x_unit_l,length(x_unit_l)),2)/sqrt(sum(direct_dtoTX.^2,2)).*1));

%L L_los

s_p=sqrt(sum(direct_RX2d.^2,2));
s=sqrt(sum(direct_d2TX.^2,2));
s_p_los=sqrt(sum(direct_RXtod.^2,2));
s_los=sqrt(sum(direct_dtoTX.^2,2));

alpha=pi/2;
n=(2*pi-alpha)/pi;
L=((s.*s_p)/(s+s_p)).*(sin(beta_0).^2);
L_los=((s_los.*s_p_los)/(s_los+s_p_los)).*(sin(beta_0_los).^2);

X=2.*beta_a.*L;
X_los=2.*beta_a.*L_los;

F=1+j/(2.*X)-3/(4.*(X.^2))-j*15/(8.*(X.^3))+75/(16.*(X.^4));
F_los=1+j/(2.*X_los)-3/(4.*(X_los.^2))-j*15/(8.*(X_los.^3))+75/(16.*(X_los.^4));

%D
alpha=pi/2;
n=(2*pi-alpha)/pi;
e0=8.854*(10^-12);
er=2.7-j*5*(10^-3)/(2*pi*f*e0); %solid reinforced concrete
gamma_h=(er.*cos(beta_0_p)-sqrt(er-sin(beta_0_p).^2))/(er.*cos(beta_0_p)+sqrt(er-sin(beta_0_p).^2));
gamma_s=(cos(beta_0_p)-sqrt(er-sin(beta_0_p).^2))/(cos(beta_0_p)+sqrt(er-sin(beta_0_p).^2));
gamma_h_los=(er.*cos(beta_0_los_p)-sqrt(er-sin(beta_0_los_p).^2))/(er.*cos(beta_0_los_p)+sqrt(er-sin(beta_0_los_p).^2))

```

```
;
gamma_s_los=(cos(beta_0_los_p)-sqrt(er-sin(beta_0_los_p)^2))/ (cos(beta_0_los_p)+sqrt(er-sin(beta_0_los_p)^2));

D1=(-exp(-j*pi/4).*cot((pi+(pi_angle-pi_angle_p))/2*n).*F)/(2.*n*sqrt(2*pi*beta_a)*sin(beta_0));
D2=(-exp(-j*pi/4).*cot((pi-(pi_angle-pi_angle_p))/2*n).*F)/(2.*n*sqrt(2*pi*beta_a)*sin(beta_0));
D3=(-exp(-j*pi/4).*cot((pi+(pi_angle+pi_angle_p))/2*n).*F)/(2.*n*sqrt(2*pi*beta_a)*sin(beta_0));
D4=(-exp(-j*pi/4).*cot((pi-(pi_angle+pi_angle_p))/2*n).*F)/(2.*n*sqrt(2*pi*beta_a)*sin(beta_0));

D1_los=(-exp(-j*pi/4).*cot((pi+(pi_angle_los-pi_angle_p_los))/2*n).*F_los)/(2.*n*sqrt(2*pi*beta_a)*sin(beta_0_los));
D2_los=(-exp(-j*pi/4).*cot((pi-(pi_angle_los-pi_angle_p_los))/2*n).*F_los)/(2.*n*sqrt(2*pi*beta_a)*sin(beta_0_los));
D3_los=(-exp(-j*pi/4).*cot((pi+(pi_angle_los+pi_angle_p_los))/2*n).*F_los)/(2.*n*sqrt(2*pi*beta_a)*sin(beta_0_los));
D4_los=(-exp(-j*pi/4).*cot((pi-(pi_angle_los+pi_angle_p_los))/2*n).*F_los)/(2.*n*sqrt(2*pi*beta_a)*sin(beta_0_los));

Ds=D1+D2+gamma_s.*(D3+D4);
Dh=D1+D2+gamma_h.*(D3+D4);

Ds_los=D1_los+D2_los+gamma_s_los.*(D3_los+D4_los);
Dh_los=D1_los+D2_los+gamma_h_los.*(D3_los+D4_los);

%diffraction strength
refl_loss2=GAMMA_r.^refl_num_d;
length2=sqrt(sum(direct_d2T X.^2,2));
delay2=length2/(3e8);
path_loss2=(1/length2).*exp(-j*beta_a*delay2);

los_length2=sqrt(sum(( repmat(TXPOINT,numofdiff_point,1)-diff_point).^2,2));
los_delay2=los_length2/(3e8);
los_loss2=(1/los_length2).*exp(-j*beta_a*los_delay2);

%streng=(sum(streng_at_d.*(-Ds-Dh).*sqrt(s_p/(s.*(s_p+s))).*exp(-j.*s).*refl_loss2.*path_loss2))/((numofdiff_point*(index_end(1,N)+1)));
%streng_los=(sum(streng_at_d_los.*(-Ds_los-Dh_los).*sqrt(s_p_los/(s_los.*(s_p_los+s_los))).*exp(-j.*s_los)).*los_loss2)/((numofdiff_point*(index_end(1,N)+1)));
%STRENG_D=(streng+streng_los);

%-----StrengDiffRefl=((streng_at_d.*(-Ds-Dh).*sqrt(s_p/(s.*(s_p+s))).*exp(-j.*s).*refl_loss2.*path_loss2))/((numofdiff_point*(index_end(1,N)+1)));
%-----StrengDiffDirec=(sum(streng_at_d_los.*(-Ds_los-Dh_los).*sqrt(s_p_los/(s_los.*(s_p_los+s_los))).*exp(-j.*s_los)).*los_loss2)/((numofdiff_point*(index_end(1,N)+1)));
StrengDiffRefl=((streng_at_d.*(-Ds-Dh).*sqrt(s_p/(s.*(s_p+s))).*exp(-j.*s).*refl_loss2.*path_loss2));
StrengDiffDirec=(sum(streng_at_d_los.*(-Ds_los-Dh_los).*sqrt(s_p_los/(s_los.*(s_p_los+s_los))).*exp(-j.*s_los)).*los_loss2);
StrengDiffRefl=repmat(StrengDiffRefl,[1,12]);
StrengDiffDirec=repmat(StrengDiffDirec,[1,12]);

DelayDiffRefl=delay1+delay2;
DelayDiffDirec=los_delay1+los_delay2;
```

<<rx_antenna.m>>

```
function F_r=rx_antenna(theta,phi,thetaT_R,phiT_R,rx_antenna_menu)
```

```
if nargin==4
    rx_antenna_menu=3;
end
```

```

global theta_tilting
global phi_tilting
global lambda

theta=theta-thetaT_R+pi/2; %NEW
%theta=theta-theta_tilting;
phi=phi-phiT_R+(1e-100)*(rx_antenna_menu>=4); %NEW;
%phi=phi-phi_tilting;

if rx_antenna_menu>=3
    omit=find((phi>pi/2)&(phi<3*pi/2));
    %----phi(omit)=nan;
    phi(omit)=0.001;
end

switch rx_antenna_menu
case 1
    F_r=1;
case 2
    beampattern=sin(theta).^3;

    D=1.6866;
    %D=1;
    F_r=D*beampattern/(max(max(beampattern)));
case 3
    % lby4 v
    kW=pi; kL=pi;
    E_phi=sin(theta) .* sin(kW/2) .* cos(theta)/cos(theta));
    AF_slot_y=2*cos(kL/2) .* sin(theta).*sin(phi));

    M=4; kdz=pi; psi_z=kdz .* cos(theta);
    S_zm=1/M .* (sin(M/2 .* psi_z)/sin(1/2 .* psi_z));
    beampattern=E_phi .*AF_slot_y .*S_zm;

    aperture=0.35*0.05;
    D=4*pi*aperture/(lambda^2);
    %D=1;
    F_r=D*beampattern/(max(max(beampattern)));
case 4
    % lby4 h
    kW=pi; kL=pi;
    E_phi=sin(theta) .* sin(kW/2) .* cos(theta)/cos(theta));
    AF_slot_y=2*cos(kL/2) .* sin(theta).*sin(phi));

    N=4; kdy=pi; psi_y=kdy .* sin(theta) .* sin(phi);
    S_yn=1/N .* (sin(N/2 .* psi_y)/sin(1/2 .* psi_y));
    beampattern=E_phi .*AF_slot_y .*S_yn;

    aperture=0.35*0.05;
    D=4*pi*aperture/(lambda^2);
    %D=1;
    F_r=D*beampattern/(max(max(beampattern)));

```

```

case 5
    %4by4
    kW=pi; kL=pi;
    E_phi=sin(theta) .* sin(kW/2 .* cos(theta)/cos(theta));
    AF_slot_y=2*cos(kL/2 * sin(theta)).*sin(phi));

    N=4; kdy=pi; psi_y=kdy * sin(theta) .* sin(phi);
    S_yn=1/N * (sin(N/2 * psi_y)/sin(1/2 * psi_y));

    M=4; kdz=pi; psi_z=kdz * cos(theta);
    S_zm=1/M * (sin(M/2 * psi_z)/sin(1/2 * psi_z));
    beampattern=E_phi .*AF_slot_y .*S_yn .*S_zm;

    aperture=0.35^2;
    D=4*pi*aperture/(lambda^2);
    %D=1;
    F_r=D*beampattern/(max(max(beampattern)));
end

<<tx_antenna.m>>

function F_t=tx_antenna(theta,phi,thetaR_T,phiR_T,tx_antenna_menu)
%F_t=tx_antenna(theta,phi,thetaR_T,phiR_T,tx_antenna_menu)

theta=theta-thetaR_T+pi/2; %NEW
phi=phi-phiR_T; %NEW

if tx_antenna_menu==1
    F_t=1;
elseif tx_antenna_menu==2
    D=1.6866;
    %D=1;
    F_t=D*sin(theta);
end

```