Jožef Stefan Institute, Ljubljana, Slovenia

Cloud-based Radio Frequency Ray Tracer: XML Application Interface v3.0

ARRS Grant No. L2-7664
JSI Technical Report
DP-12586

Dr. Roman Novak
Ljubljana, 14.2.2018

Contents

About the report

Activity A2.3 of the project Advanced Ray-Tracing Techniques in Radio Environment Characterization and Radio Localization was primarily concerned with the design and implementation of a well-defined application interface (API) for cloud-based radio frequency ray tracer. The outcome of this activity is the fully defined and implemented programmatic access towards the services in the cloud. We were targeting REST-like interface, which is one of the most widely used protocols. The API covers the input parameters, including environment geometry, selected materials, antenna characteristics, algorithm parameters, such as types of rays and their ranges, as well as the output metrics.

This document provides the XML API version 3.0. The description of the request and response XMLs is organized as a reference manual. Two versions of response XML are foreseen: signal loss and channel impulse response (CIR). The first provides signal loss results at multiple observation points on a plane, whereas the second lists multipath components of narrowband CIR at single observation point.

The project is supported by the Slovenian Research Agency under Grant No. L2-7664. It takes place at the Department of Communication Systems in collaboration with Alanta and Xlab.

Elements of Request XML

config

Config element is a container of all request elements. It embeds information about the transmitter, observation plane, ray-tracing method and algorithm parameters, as well as the description of the entire scene geometry.

Parents

none

Children

<> version	Request specification version
<> request_id	32-bit request identification
<> description	Name or some other text describing the request
<> transmitter	Transmitter properties
<> receivers	Observation plane specification including single CIR evaluation point
<> method	Ray generation method
<> subdivision	Icosahedral grid subdivision parameters
<> raytracing	Ray-tracing parameters
<> architecture	Scene geometry

Attributes

none

Example Syntax

<config> ... </config>

version

The request specification version

Parents

<> config

Children

none

Attributes

none

Example Syntax

<version> 3.0 </version>

request_id

Each request should contain 32-bit identification. The identification is copied unchanged to the response. It may be used by the client to match responses and requests.

Parents

<> config

Children

none

Attributes

none

Example Syntax

<request_id> 1958199045 </request_id>

description

Description of the request should be given as plain text. Any description may be entered, including XML tags, by using CDATA section syntax.

Parents

<> config

Children

none

Attributes

none

Example Syntax

<description> <![CDATA[Simple CIR]]> </description>

transmitter

Element describes properties of the transmitting antenna, including central frequency wavelength, antenna position and orientation, the transmission power and the requested radiation pattern.

Parents

<> config

Children

<> wavelength	Wavelength of the central frequency in meters
<> position	Absolute position of the transmitting antenna in space
<> direction	Unit vector of dipole orientation
<> isotropic	Transmitting equally to all directions (1) or by applying dipole pattern (0)
<> power	Transmitting power in watts

Attributes

none

Example Syntax

<transmitter> ... </transmitter>

wavelength

Wavelength of the transmission frequency entered in meters. The value should not significantly exceed the wavelength of materials database, which was measured at 1.8 GHz (0.17m).

Parents

<> transmitter

Children

none

Attributes

none

Example Syntax

<wavelength> 0.1225 </wavelength>

position

Position is a triplet of x, y and z coordinates, representing absolute position in space. Values should be entered in meters. Position is used in two contexts. As a child of *transmitter* element it defines location of the transmitting antenna. When used in the *receivers* element, it gives a point on the observation plane. If computation of CIR is requested by setting *ppm* element to 0, the position is the location of CIR measurement.

Parents

<> transmitter
<> receivers

Children

none

Attributes

none

Example Syntax

<position> 2.00000000 2.00000000 1.00000000 </position>

direction

Direction is a triplet of x, y and z coordinates, describing antenna dipole orientation. Unit vector is expected. If isotropic antenna is requested, then *direction* has no particular meaning.

Parents

<> transmitter
<> receivers

Children

none

Attributes

none

Example Syntax

<direction> 0.00000000 0.00000000 1.00000000 </direction>

isotropic

Antennas are configured to evenly distribute power in all directions if *isotropic* flag is set. Otherwise antennas act as ideal dipoles.

Parents

<> transmitter
<> receivers

Children

none

Attributes

none

Example Syntax

<isotropic> 0 </isotropic>

power

Power value defines the transmission power in watts. It is present only for compatibility reasons. Ray tracer does not make any use of this value.

Parents

<> transmitter

Children

none

Attributes

none

Example Syntax

<power> 0.10000000 </power>

receivers

Element describes the signal observation plane and receivers used to sample signal on that plane. Reception antenna orientation and radiation pattern apply to all sampling points. The observation plane has the shape of a rectangular area with given density of sampling points.

Parents

<> config

Children

<> area	Description of the signal observation plane
<> position	Absolute position of a point on the observation plane
<> direction	Unit vector of dipole orientation
<> isotropic	Receiving equally from all directions (1) or as an ideal dipole (0)

Attributes

none

Example Syntax

<receivers> ... </receivers>

area

Element gives geometric position of the signal observation plane and the density of receivers on this plane at which signal is to be evaluated.

Parents

<> receivers

Children

<> dimension	Width and height of the rectangular area
<> rotation	Rotation of the rectangular area around orthogonal axes
<> translation	Translation of the rotated area
<> ppm	Evaluation points per meter for both dimensions

Attributes

none

Example Syntax

<area> ... </area>

dimension

Dimension element is two or three component vector giving object dimensions in meters, either in 2D or 3D. Components correspond to x, y and optionally z directions, respectively.

Parents

<> area
<> wall
<> window
<> door
<> hole

Children

none

Attributes

none

Example Syntax

<dimension> 11.51000000 3.44000000 </dimension>
<dimension> 11.51000000 0.50000000 3.44000000 </dimension>

rotation

Rotation element is a single or three component vector specifying rotation angles in radians. *Hole* element has only one possible axis of rotation, which is perpendicular to the *Wall* element main surface. *Area* elements and *Wall* elements may be rotated around any of the orthogonal axes x, y, and z. Rotation matrix derived from the vector is intrinsic Tait-Bryan type x-y-z matrix, where axes of the rotating coordinate system move with the object.

Parents

<> area
<> wall
<> window
<> door
<> hole

Children

none

Attributes

none

Example Syntax

<rotation> 0.00000000 </rotation>
<rotation> 1.57079633 0.00000000 0.00000000 </rotation>

translation

Translation element is a vector of two or three components describing object translation. While *Hole* element is translated in two dimensions on a plane defined by the parent *Wall*, full 3D translation applies to *Area* and *Wall* elements.

Parents

<> area
<> wall

<> window
<> door
<> hole

Children

none

Attributes

none

Example Syntax

<translation> 6.11000000 9.50000000 </translation>
<translation> -0.83333333 -3.16666667 0.85000000 </translation>

ppm

Points per meter (*ppm*) element defines density of evaluation points on the observation plane. Density applies to one dimension, e.g., 10 points per meter gives 100 points per square meter.

Parents

<> area

Children

none

Attributes

none

Example Syntax

<ppm> 30.00000000 </ppm>

method

Currently supported method is shooting and bouncing rays (SBR) while using recursive icosahedral grids as initial ray launching template. Thus, the only acceptable value of the method element is 0. Other methods are planned for future releases.

Parents

<> config

Children

none

Attributes

none

Example Syntax

<method> 0 </method>

subdivision

Subdivision element effectively defines the total number of rays launched from the transmitter, as further explained in the *depth* element reference. Next, the granularity of simulation steps is specified, affecting the percentage reporting interval.

Parents

<> config

Children

<> depth	Icosahedron subdivision depth
<> rays_per_step	Number of rays per step

Attributes

none

Example Syntax

<subdivision> ... </subdivision>

depth

Depth element within *subdivision* element defines the icosahedral grid refinement level for the recursive grid construction algorithm [1]. Given level n a total of $10 \times 2^{2n} + 2$ rays are launched from the transmitter. The total number of rays must be sufficient to guarantee at least one hit per reception sphere around any of the observation points, taking into account the allowed number of interactions, the maximum size of a sphere and the size of environment geometry.

Parents

<> subdivision

Children

none

Attributes

none

Example Syntax

<depth> 10 </depth>

rays_per_step

Number of rays that are traced as a single step has a role of optimization parameter. GPU load balancing is dependent on this parameter. A value that gives shortest running time should be used.

Parents

<> subdivision

Children

none

Attributes

none

Example Syntax

<rays_per_step> 40962 </rays_per_step>

raytracing

Raytracing element embeds tracing algorithm parameters.

Parents

<> config

Children

<> depth	Ray-tracing depth, i.e., number of consecutive interactions
<> accumulate	Signal accumulation components
<> rx_radius	Reception sphere radius
<> diffraction	Diffraction parameters
<> bloom	Bloom filter parameters
<> power_limit	Stopping criterion in addition to ray-tracing depth
<> cir_entries	Maximum number of channel impulse response taps

Attributes

none

Example Syntax

<raytracing> ... </raytracing>

depth

Depth element within *raytracing* element sets limit on the number of consecutive interactions (reflections, refractions, diffractions). Ray is terminated sooner if *power_limit* is reached.

Parents

<> raytracing

Children

none

Attributes

none

Example Syntax

<depth> 30 </depth>

accumulate

Accumulate element contains a bitmask specifying accumulated types of rays at observation points: 1-line-of-sight rays, 2-reflected rays, 4-refracted rays, 8-diffracted rays. Typical bitmask values are 7 and 15.

Parents

<> raytracing

Children

none

Attributes

none

Example Syntax

<depth> 7 </depth>

rx_radius

Reception spheres are used to detect rays passing by observation points. Sphere should have non-zero radius that guarantee one hit per wavefront. The value depends on the number of initial rays, ray-tracing depth and on the size of environment geometry.

Parents

<> raytracing

Children

none

Attributes

none

Example Syntax

<rx_radius> 0.01500000 </rx_radius>

diffraction

Diffraction element embeds parameters that are specific for diffraction events. Currently only a radius of edge cylinders can be set.

Parents

<> raytracing

Children

<> edge_radius	Radius of cylinders around diffraction edges

Attributes

none

Example Syntax

<diffraction> ... </diffraction>

edge_radius

Diffraction is modelled using Uniform Theory of Diffraction (UTD) [2]. Each ray passing in close vicinity to diffraction edge triggers a number of rays within Keller cone. In order to detect nearby rays, a similar approach to reception spheres is used. The size of a radius depends on the number of initial rays, ray-tracing depth and on the size of environment geometry.

Parents

<> diffraction

Children

none

Attributes

none

Example Syntax

<edge_radius> 0.01500000 </edge_radius>

bloom

Bloom filter is used for double counting avoidance [1]. The element contains filter parameters that need to be set for efficient reduction of algorithmic errors.

Parents

<> raytracing

Children

<> k	Number of independent hash functions
<> m	Filter size in bits rounded up to a 32-bit boundary

Attributes

none

Example Syntax

<bloom> ... </bloom>

k

The element gives the number of independent hash functions used for Bloom filtering [1]. Given allowed false positive rate p, the appropriate value of k is $-\log_2 p$.

Parents

<> bloom

Children

none

Attributes

none

Example Syntax

<k> 14 </k>

m

Filter size *m* gives the number of bits for filter storage [1]. It should be rounded up to a multiple of 32 bits. Given false probability rate p, maximal number of expected wavefronts n at any observation point and optimal number of hash functions k the required size is -n ln p / (ln 2 ln 2).

Parents

<> bloom

Children

none

Attributes

none

Example Syntax

<m> 288 </m>

cir_entries

The element value should be set to 1 when signal loss is evaluated at multiple observation points. In case of CIR evaluation in a single point, it should be set to at least maximum number of different wavefronts expected at given observation point, or equivalently, the number of taps in the expected channel impulse response. If the value is too small, extra taps will be removed from the result, while exceedingly large numbers will unnecessary consume simulation memory.

Parents

<> raytracing

Children

none

Attributes

none

Example Syntax

<cir_entries> 1000 </cir_entries>

power_limit

The element defines two parameters used to limit ray tracing to acceptable signal loss.

Parents

<> raytracing

Children

<> max_amp	Constant related to maximum loss
<> max_d_to_a	Constant giving acceptable ratio between distance and attenuation

Attributes

none

Example Syntax

<power_limit> ... </power_limit>

max_amp

Constant is used in stopping a ray as soon as signal loss of the path is larger than given threshold. The constant should be evaluated as (4*pi/0.0749)^2*10^(-(lossdB+4.3)/10).

Parents

<> power_limit

Children

none

Attributes

none

Example Syntax

<max_amp> 3.909700e-009 </max_amp>

max_d_to_a

Second constant used in stopping rays gives acceptable ratio of distance to attenuation. It should be calculated as 0.0749/4/pi*10^((lossdB+4.3)/20).

Parents

<> power_limit

Children

none

Attributes

none

Example Syntax

<max_d_to_a> 1.599300e+004 </max_d_to_a>

architecture

This is main container of simulated environment geometry, including some visualization helpers.

Parents

<> config

Children

<> scene	Scene description

Attributes

none

Example Syntax

<architecture> ... </architecture>

scene

The entire scene is described entirely as block-like walls, windows and doors.

Parents

<> architecture

Children

<> wall	Wall description
<> window	Window description
<> door	Door description

Attributes

none

Example Syntax

<scene> ... </scene>

wall

Wall is a block-like object consisting of uniform material and, optionally, having a number of holes or openings.

Parents

<> scene

Children

<> dimension	Wall dimension
<> rotation	Wall rotation
<> translation	Wall translation
<> material	Wall material
<> hole	Set of holes
<> surfaces	Surfaces description
<> edges	Diffraction edges
<> texture	Visualization texture
<> tex_height	Texture scaling
<> separating_floors	Floor separation flag
<> floor	Floor tagging

Attributes

none

Example Syntax

<wall> ... </wall>

material

Material is a vector of two indices. The first index gives material group, following by the index within that group. Materials database is set by materials.xml. For each material, permittivity, permeability and conductivity at 1.8 GHz are specified.

Parents

<> wall
<> window
<> door

Children

none

Attributes

none

Example Syntax

<material> 2 1 </material>

hole

There may be multiple *hole* elements within an object. Hole is a rectangular opening through the object. The direction of a hole is always from one main side to the other. Holes may overlap each other and object boundaries, giving complex final shapes.

Parents

<> wall
<> window
<> door

Children

<> dimension	Hole dimension
<> rotation	Hole rotation in space
<> translation	Hole translation within an object

Attributes

none

Example Syntax

<hole> ... </hole>

surfaces

Surfaces element embeds description of object's sides in the form of tessellated triangles.

Parents

<> wall
<> window
<> door

Children

<> surface	Single surface

Attributes

none

Example Syntax

<surfaces> ... </surfaces>

surface

Each surface is described as a list of points in space, with consecutive triplets interpreted as triangles. Triangle normals should point out of the described object. Largest (main) surfaces should always have indices 0 and 1.

Parents

<> surfaces

Children

<> point	Triangle point in space

Attributes

id	Surface index, zero based

Example Syntax

<surface id="0"> ... </surface>

point

Point element as part of the triangle description is a triplet of x, y and z coordinates, representing absolute position in space. If outer edge flag is set, it indicates that the triangle side starting in this point is one of side's outer edges.

Parents

<> surface

Children

none

Attributes

e — Outer edge flag

Example Syntax

<point e="1"> 16.58000000 0.00000000 0.00000000 </point>

edges

Edges element embeds description of object's diffraction triggering sides.

Parents

<> wall
<> window
<> door

Children

<> edge — Single edge

Attributes

none

Example Syntax

<edges> ... </edges>

edge

Diffraction edge is described as an ordered tuple of two consecutive *point* elements and two consecutive *side* elements. Edge point is either starting or ending edge vertex, whereas sides are unit vectors describing direction of opaque sides starting from the edge. By definition, anti-clockwise rotation of the first side vector into the second vector using the edge as the rotation axis describes object's diffraction space.

Parents

<> edges

Children

<> point	Edge vertex in space
<> side	Edge side direction

Attributes

none

Example Syntax

<edge> ... </edge>

point

Point element as part of edge description is a triplet of x, y and z coordinates, representing absolute position in space.

Parents

<> edge

Children

none

Attributes

none

Example Syntax

<point> 2.50000000 -4.85000000 5.00000000 </point>

side

Side element as part of edge description is a triplet of x, y and z coordinates, representing unit direction vector.

Parents

<> edge

Children

none

Attributes

none

Example Syntax

<side> -1.00000000 0.00000000 0.00000000 </side>

texture

Texture index is not used by cloud server. It is used by client for visualization purposes.

Parents

<> wall
<> window
<> door

Children

none

Attributes

none

Example Syntax

<texture> 0 </texture>

tex_height

Tex_height element is not used by cloud server. It is used by client for visualization purposes.

Parents

<> wall
<> window
<> door

Children

none

Attributes

none

Example Syntax

<tex_height> 1.00000000 </tex_height>

separating_floors

Separating_floors element is not used by cloud server. It is used by client for visualization purposes.

Parents

<> wall
<> window
<> door

Children

none

Attributes

none

Example Syntax

<separating_floors> 0 </separating_floors>

floor

Floor element is not used by cloud server. It is used by client for visualization purposes.

Parents

<> wall
<> window

<> door

Children

none

Attributes

none

Example Syntax

<floor> 0 </floor>

window

Window is a block-like object consisting of uniform material, usually glass, and, optionally having a number of holes or openings.

Parents

<> scene

Children

<> dimension	Window dimension
<> rotation	Window rotation
<> translation	Window translation
<> material	Window material
<> hole	Set of holes
<> surfaces	Surfaces description
<> edges	Diffraction edges
<> texture	Visualization texture
<> tex_height	Texture scaling
<> separating_floors	Floor separation flag
<> floor	Floor tagging

Attributes

none

Example Syntax

<window> ... </window>

door

Door is a block-like object consisting of uniform material, usually wood, and, optionally having a number of holes or openings.

Parents

<> scene

Children

<> dimension	Door dimension
<> rotation	Door rotation
<> translation	Door translation
<> material	Door material
<> hole	Set of holes
<> surfaces	Surfaces description
<> edges	Diffraction edges
<> texture	Visualization texture
<> tex_height	Texture scaling
<> separating_floors	Floor separation flag
<> floor	Floor tagging

Attributes

none

Example Syntax

<door> ... </door>

Elements of Loss Response XML

signal

Element is a container of signal loss response elements. It embeds information about the request, computation time and signal loss at the observation points.

Parents

none

Children

<> request_id	32-bit request identification
<> preproc_time	Scene pre-processing time in seconds
<> gpu_time	GPU ray-tracing time in seconds
<> loss_db	Signal loss at observation points in dB

Attributes

none

Example Syntax

<signal> ... </signal>

request_id

Each request is identified by 32-bit number. The identification is copied unchanged from the request XML. It may be used by the client to match responses and requests.

Parents

<> signal

Children

none

Attributes

none

Example Syntax

<request_id> 1958199045 </request_id>

preproc_time

Element contain the total scene pre-processing time in seconds.

Parents

<> signal

Children

none

Attributes

none

Example Syntax

<preproc_time> 0.41500000 </preproc_time>

gpu_time

Element contain the total GPU computing time in seconds.

Parents

<> signal

Children

none

Attributes

none

Example Syntax

<gpu_time> 7.36100000 </gpu_time>

loss_db

Element embeds the signal loss values at observation points.

Parents

<> signal

Children

<> num_x	Number of observation points in x direction
<> num_y	Number of observation points in y direction
<> rx	Signal loss

Attributes

none

Example Syntax

<loss_db> ... </loss_db>

num_x

Element defines the number of observation points in x direction.

Parents

<> loss_db

Children

none

Attributes

none

Example Syntax

<num_x> 1000 </num_x>

num_y

Element defines the number of observation points in y direction.

Parents

<> loss_db

Children

none

Attributes

none

Example Syntax

<num_y> 1000 </num_y>

rx

Calculated signal loss for each observation point

Parents

<> loss_db

Children

<> y	Single row of signal loss values

Attributes

none

Example Syntax

<rx> ... </rx>

y

Single row of signal loss values is embedded within y element. Rows with increasing y coordinate are listed consecutively.

Parents

<> rx

Children

<> x Calculated signal loss in dB

Attributes

none

Example Syntax

<y> … </y>

x

Calculated signal loss in dB at implicitly defined observation point. Absolute coordinates of the observation point should be derived from the observation area geometry, from the density of observation points and from the ordering of reported elements.

Parents

<> y

Children

none

Attributes

none

Example Syntax

<x> 121.42367115 </x>

Elements of CIR Response XML

cir

Cir element is a container of narrowband CIR response. It embeds information about the request, computation time and narrowband CIR taps at single observation point. Each tap represents a multipath component of polarity sign-extended real amplitude, multiplied by time delayed Dirac-Delta function. The reported taps are not sorted in any way.

Parents

none

Children

<> request_id	32-bit request identification
<> preproc_time	Scene pre-processing time in seconds
<> gpu_time	GPU ray-tracing time in seconds
<> taps	Multipath components

Attributes

none

Example Syntax

<cir> ... </cir>

request_id

Each request is identified by 32-bit number. The identification is copied unchanged from the request XML. It may be used by the client to match responses and requests.

Parents

<> cir

Children

none

Attributes

none

Example Syntax

<request_id> 1958199045 </request_id>

preproc_time

Element contain the total scene pre-processing time in seconds.

Parents

<> cir

Children

none

Attributes

none

Example Syntax

<preproc_time> 0.41500000 </preproc_time>

gpu_time

Element contain the total GPU computing time in seconds.

Parents

<> cir

Children

none

Attributes

none

Example Syntax

<gpu_time> 7.36100000 </gpu_time>

taps

Element embeds the narrowband CIR taps in no particular order.

Parents

<> cir

Children

<> num	Total number of taps
<> tap	Tap record

Attributes

none

Example Syntax

<taps> ... </taps>

num

The total number of multipath components is always the first element of taps.

Parents

<> taps

Children

none

Attributes

none

Example Syntax

<num> 12304 </num>

tap

Tap element contains a record describing single multipath component.

Parents

<> taps

Children

<> delay	Propagation path delay in ns
<> Re	Real component of electric field phasor
<> Im	Imaginary component of electric field phasor
<> refl	Number of propagation path reflections
<> refr	Number of propagation path refractions
<> diff	Number of propagation path diffractions
<> air	Propagation distance in air
<> mat	Propagation distance within non-air material
<> L	Polarisation mismatch

Attributes

none

Example Syntax

<tap> ... </tap>

delay

Propagation path delay given in ns

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<delay> 48.19524690 </delay>

Re

Element contains the real component of the electric field phasor assuming 1W transmission.

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<Re> 3.20791441e-004 </Re>

Im

Element contains the imaginary component of the electric field phasor assuming 1W transmission.

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<Im> -1.09798668e-004 </Im>

refl

The total number of reflections encountered on the propagation path is provided as debugging information.

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<refl> 5 </refl>

refr

The total number of refractions encountered on the propagation path is provided as debugging information.

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<refr> 7 </refr>

diff

The total number of edge diffractions encountered on the propagation path is provided as debugging information. Currently only a single diffraction is supported.

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<refl> 1 </refl>

air

Element gives the total propagation distance in air.

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<air> 18.43352318 </air>

mat

Element gives the total propagation distance in non-air medium.

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<mat> 0.30179408 </mat>

L

Polarisation mismatch at receiving antenna can be used to derive received pulse polarity.

Parents

<> tap

Children

none

Attributes

none

Example Syntax

<L> 0.99999976 </L>

References

[1] R. Novak, "Bloom filter for double counting avoidance in radio frequency ray tracing," submited for publication, Jan 2018.

[2] D. A. McNamara, C. W. I. Pistorius, J. A. G. Malherbe, "Introduction to the uniform geometrical theory of diffraction." Norwood, MA: Artech House, Antennas and Propagation Library, 1990.

Acronyms

CIR	Channel Impulse Response
GPU	Graphical Processing Unit
UTD	Uniform Theory of Diffraction
XML	eXtensible Markup Language

Index