

Symulacja komputerowa zaniku sygnału luminescencyjnego w skaleniach

Generated by Doxygen 1.8.12

Contents

1	Symulacja komputerowa zaniku sygnału luminescencyjnego w skaleniach	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	Crystal Class Reference	5
3.1.1	Detailed Description	6
3.1.2	Constructor & Destructor Documentation	6
3.1.2.1	Crystal()	6
3.1.3	Member Function Documentation	6
3.1.3.1	calculateDistance()	6
3.1.3.2	calculateTau()	6
3.1.3.3	changeTime()	7
3.1.3.4	countElectrons()	7
3.1.3.5	saveToFile()	7
3.1.3.6	startSimulation()	8
3.1.3.7	tunnelEffect()	8
3.1.3.8	tunnelEffectProbability()	8
3.2	Electron Class Reference	9
3.2.1	Detailed Description	9
3.2.2	Constructor & Destructor Documentation	9
3.2.2.1	Electron()	9
3.2.3	Member Function Documentation	10

3.2.3.1	setX()	10
3.2.3.2	setY()	10
3.2.3.3	setZ()	10
3.3	ElectronHole Class Reference	10
3.3.1	Detailed Description	11
3.3.2	Constructor & Destructor Documentation	11
3.3.2.1	ElectronHole()	11
3.3.3	Member Function Documentation	12
3.3.3.1	getEnergy()	12
3.3.3.2	getTrap()	12
3.3.3.3	getX()	12
3.3.3.4	getY()	12
3.3.3.5	getZ()	12
3.4	Trap Class Reference	13
3.4.1	Detailed Description	13
3.4.2	Constructor & Destructor Documentation	13
3.4.2.1	Trap()	13
3.4.3	Member Function Documentation	14
3.4.3.1	getElectron()	14
3.4.3.2	getEnergy()	14
3.4.3.3	getX()	14
3.4.3.4	getY()	14
3.4.3.5	getZ()	14
3.4.3.6	isOccupied()	14
3.4.3.7	removeElectron()	14
3.4.3.8	setElectron()	15
	Index	17

Chapter 1

Symulacja komputerowa zaniku sygnału luminescencyjnego w skaleniach

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Crystal	5
Electron	9
ElectronHole	10
Trap	13

Chapter 3

Class Documentation

3.1 Crystal Class Reference

```
#include <Crystal.h>
```

Public Member Functions

- [Crystal](#) (long long int n_el, long long n_holes, double min, double max)
- double [changeTime](#) (unsigned long time) const
- void [startSimulation](#) (int time)
- void [tunnelEffect](#) ([Trap](#) &trap, int time)
- unsigned long [countElectrons](#) () const
funkcja zliczająca ilość elektronów w pułapkach
- void [saveToFile](#) (std::string)
zapis wyniku do plików

Static Public Member Functions

- static double [tunnelEffectProbability](#) (double time, double tau)

Private Member Functions

- double [calculateTau](#) (double distance, [ElectronHole](#) *el_hole, const [Trap](#) &trap) const
- double [calculateDistance](#) (const [Trap](#) &trap, const [ElectronHole](#) *hole) const

Private Attributes

- const double [S](#) = 3e15
wartość stałej 'attempt-to-escape frequency'
- std::map< unsigned long, unsigned long > [amount_electrons](#)
mapa klucz - czas, wartosc - ilosc elektronow w stanie wzbudzonym
- std::vector< [Trap](#) > [el_traps](#)
wektor przechowujący wszystkie pułapki
- std::vector< [Electron](#) * > [electrons](#)
wektor przechowujący wskaźniki do elektronów
- std::vector< [ElectronHole](#) * > [electron_holes](#)
wektor przechowujący wskaźniki do dziur elektronowych

3.1.1 Detailed Description

Klasa reprezentująca kryształ

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Crystal()

```
Crystal::Crystal (
    long long int n_el,
    long long n_holes,
    double min,
    double max )
```

Konstruktor

Parameters

<i>n_{el}</i>	ilość obiektów elektronu do stworzenia
<i>n_{el}</i>	ilość obiektów dziury elektronowej do stworzenia
<i>min</i>	dolna granica którą mogą przyjmować współrzędne cząstek
<i>max</i>	górną granicą którą mogą przyjmować współrzędne cząstek

3.1.3 Member Function Documentation

3.1.3.1 calculateDistance()

```
double Crystal::calculateDistance (
    const Trap & trap,
    const ElectronHole * hole ) const [private]
```

Funkcja do obliczania odległości między pułapką a dziurą

Parameters

<i>trap</i>	referencja do pułapki
<i>hole</i>	wskaznik na dziurę elektronową

Returns

odległość między parametrami

3.1.3.2 calculateTau()

```
double Crystal::calculateTau (
    double distance,
```

```
ElectronHole * el_hole,  
const Trap & trap ) const [private]
```

Funkcja do obliczania wartosci tau

See also

[calculateDistance\(\)](#)

Parameters

<i>distance</i>	dystans do przetunelowania
<i>el_hole</i>	wskaznik na dziurę elektronową
<i>trap</i>	referencja do pułapki

Returns

wartość tau

3.1.3.3 changeTime()

```
double Crystal::changeTime (  
    unsigned long time ) const
```

Funkcja zmieniająca jednostkę czasu

Parameters

<i>time</i>	czas do zamiany
-------------	-----------------

Returns

czas podany w jednostce $\log_{10}(t/2\text{dni})$

3.1.3.4 countElectrons()

```
unsigned long Crystal::countElectrons ( ) const
```

funkcja zliczająca ilość elektronów w pułapkach

Returns

ilość elektronów w pułapkach

3.1.3.5 saveToFile()

```
void Crystal::saveToFile (  
    std::string name )
```

zapis wyniku do plików

Parameters

<i>name</i>	nazwa pliku wyjściowego
-------------	-------------------------

3.1.3.6 startSimulation()

```
void Crystal::startSimulation (
    int time )
```

Funkcja rozpoczynająca symulację

Parameters

<i>time</i>	symulowany czas działania
-------------	---------------------------

3.1.3.7 tunnelEffect()

```
void Crystal::tunnelEffect (
    Trap & trap,
    int time )
```

Funkcja wykonująca efekt tunelowania

Parameters

<i>trap</i>	referencja do pułapki
<i>time</i>	czas

3.1.3.8 tunnelEffectProbability()

```
double Crystal::tunnelEffectProbability (
    double time,
    double tau ) [static]
```

Funkcja do obliczania prawdopodobieństwa NIEZAJŚCIA tunelowania

Parameters

<i>time</i>	czas
<i>tau</i>	wartość tau

See also

[calculateTau\(\)](#)

Returns

wartość prawdopodobieństwa

The documentation for this class was generated from the following files:

- C:/Users/olav/ClionProjects/Dissertation/Crystal.h
- C:/Users/olav/ClionProjects/Dissertation/Crystal.cpp

3.2 Electron Class Reference

```
#include <Electron.h>
```

Public Member Functions

- [Electron](#) (std::vector< double > pos)
- void [setX](#) (double x)
- void [setY](#) (double y)
- void [setZ](#) (double z)

Private Attributes

- std::vector< double > [position](#)
wektor współrzędnych elektronu

Friends

- std::ostream & **operator**<< (std::ostream &s, const [Electron](#) &v)

3.2.1 Detailed Description

Klasa reprezentująca elektron

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Electron()

```
Electron::Electron (  
    std::vector< double > pos )
```

Konstruktor tworzy obiekt o podanych współrzędnych

Parameters

<i>pos</i>	wektor współrzędnych
------------	----------------------

3.2.3 Member Function Documentation

3.2.3.1 setX()

```
void Electron::setX (
    double x )
```

Zmienna x-owej wartości współrzędnej

Parameters

x	nowa wartość współrzędnej
---	---------------------------

3.2.3.2 setY()

```
void Electron::setY (
    double y )
```

Zmienna y-owej wartości współrzędnej

Parameters

y	nowa wartość współrzędnej
---	---------------------------

3.2.3.3 setZ()

```
void Electron::setZ (
    double z )
```

Zmienna z-owej wartości współrzędnej

Parameters

z	nowa wartość współrzędnej
---	---------------------------

The documentation for this class was generated from the following files:

- C:/Users/olav/ClionProjects/Dissertation/Electron.h
- C:/Users/olav/ClionProjects/Dissertation/Electron.cpp

3.3 ElectronHole Class Reference

```
#include <ElectronHole.h>
```

Public Member Functions

- [ElectronHole](#) (std::vector< double > pos, [Trap](#) &trap)
- double [getEnergy](#) () const
- double [getX](#) () const
- double [getY](#) () const
- double [getZ](#) () const
- [Trap](#) * [getTrap](#) ()
- void [nullTrap](#) ()
usuwa dziurę z pułapki, ustawia wskaźnik trap na NULL
- [~ElectronHole](#) ()
destruktor

Private Attributes

- std::vector< double > [position](#)
wektor współrzędnych dziury
- [Trap](#) * [trap](#) = NULL
wskaźnika na obiekt typu [Trap](#) (informacja czy obiekt znajduje się w pułapce)
- double [energy](#) = 1.
energia dziury [w eV]

Friends

- std::ostream & **operator**<< (std::ostream &s, const [ElectronHole](#) &v)

3.3.1 Detailed Description

Klasa reprezentująca dziurę elektronową

3.3.2 Constructor & Destructor Documentation

3.3.2.1 ElectronHole()

```
ElectronHole::ElectronHole (
    std::vector< double > pos,
    Trap & trap )
```

Konstruktor tworzy obiekt o podanych współrzędnych i łączy go z pułapką

Parameters

<i>pos</i>	wektor współrzędnych
<i>trap</i>	referencja do pułapku

3.3.3 Member Function Documentation

3.3.3.1 getEnergy()

```
double ElectronHole::getEnergy ( ) const
```

Returns

zwraca energię dziury

3.3.3.2 getTrap()

```
Trap* ElectronHole::getTrap ( )
```

Returns

zwraca adres do pułapki w której się obecnie znajduje

3.3.3.3 getX()

```
double ElectronHole::getX ( ) const
```

Returns

zwraca x-ową współrzędną

3.3.3.4 getY()

```
double ElectronHole::getY ( ) const
```

Returns

zwraca y-ową współrzędną

3.3.3.5 getZ()

```
double ElectronHole::getZ ( ) const
```

Returns

zwraca z-ową współrzędną

The documentation for this class was generated from the following files:

- C:/Users/olav/ClionProjects/Dissertation/ElectronHole.h
- C:/Users/olav/ClionProjects/Dissertation/ElectronHole.cpp

3.4 Trap Class Reference

```
#include <Trap.h>
```

Public Member Functions

- `Trap` (`std::vector< double > position`)
- `double getEnergy () const`
- `double getX () const`
- `double getY () const`
- `double getZ () const`
- `void setElectron (Electron *electron1)`
- `Electron * getElectron () const`
- `void removeElectron (std::vector< double > position)`
- `bool isOccupied () const`

Private Attributes

- `std::vector< double > position`
wektor współrzędnych pułapki
- `Electron * electron = NULL`
wskaznik na uwięziony elektron
- `double energy = 2.`
energia pułapki [w eV]

Friends

- `std::ostream & operator<< (std::ostream &s, const Trap &v)`

3.4.1 Detailed Description

Klasa reprezentująca pułapkę

3.4.2 Constructor & Destructor Documentation

3.4.2.1 Trap()

```
Trap::Trap (  
    std::vector< double > position )
```

Konstruktor tworzy obiekt o podanych współrzędnych

Parameters

<code>position</code>	wektor współrzędnych
-----------------------	----------------------

3.4.3 Member Function Documentation

3.4.3.1 getElectron()

```
Electron * Trap::getElectron ( ) const
```

Returns

wskaźnik na elektron znajdujący się w pułapce

3.4.3.2 getEnergy()

```
double Trap::getEnergy ( ) const
```

Returns

zwraca energię pułapki

3.4.3.3 getX()

```
double Trap::getX ( ) const
```

Returns

zwraca x-ową współrzędną

3.4.3.4 getY()

```
double Trap::getY ( ) const
```

Returns

zwraca y-ową współrzędną

3.4.3.5 getZ()

```
double Trap::getZ ( ) const
```

Returns

zwraca z-ową współrzędną

3.4.3.6 isOccupied()

```
bool Trap::isOccupied ( ) const
```

sprawdza czy w pułapce znajduje się elektron

Returns

TRUE jeśli elektron jest spułapkowany

3.4.3.7 removeElectron()

```
void Trap::removeElectron (
    std::vector< double > position )
```

Usuwa elektorn z pułapki

Parameters

<i>position</i>	nowa pozycja elektronu
-----------------	------------------------

3.4.3.8 setElectron()

```
void Trap::setElectron (
    Electron * electron1 )
```

pułapkuje elektron, ustawia wskaźnik na niego

Parameters

<i>electron1</i>	elektron do spułapkowania
------------------	---------------------------

The documentation for this class was generated from the following files:

- C:/Users/olav/ClionProjects/Dissertation/Trap.h
- C:/Users/olav/ClionProjects/Dissertation/Trap.cpp

Index

calculateDistance
 Crystal, [6](#)
calculateTau
 Crystal, [7](#)
changeTime
 Crystal, [7](#)
Crystal, [5](#)
 calculateDistance, [6](#)
 calculateTau, [7](#)
 changeTime, [7](#)
 Crystal, [6](#)
 startSimulation, [7](#)
 tunnelEffect, [8](#)
 tunnelEffectProbability, [8](#)

Electron, [9](#)
 Electron, [9](#)
 setX, [9](#)
 setY, [10](#)
 setZ, [10](#)
ElectronHole, [10](#)
 ElectronHole, [11](#)
 getEnergy, [11](#)
 getTrap, [11](#)
 getX, [11](#)
 getY, [12](#)
 getZ, [12](#)
 nullTrap, [12](#)

getEnergy
 ElectronHole, [11](#)
 Trap, [13](#)
getTrap
 ElectronHole, [11](#)
getX
 ElectronHole, [11](#)
 Trap, [13](#)
getY
 ElectronHole, [12](#)
 Trap, [13](#)
getZ
 ElectronHole, [12](#)
 Trap, [14](#)

isOccupied
 Trap, [14](#)

nullTrap
 ElectronHole, [12](#)

removeElectron

 Trap, [14](#)
setElectron
 Trap, [14](#)
setX
 Electron, [9](#)
setY
 Electron, [10](#)
setZ
 Electron, [10](#)
startSimulation
 Crystal, [7](#)
Trap, [12](#)
 getEnergy, [13](#)
 getX, [13](#)
 getY, [13](#)
 getZ, [14](#)
 isOccupied, [14](#)
 removeElectron, [14](#)
 setElectron, [14](#)
 Trap, [13](#)
tunnelEffect
 Crystal, [8](#)
tunnelEffectProbability
 Crystal, [8](#)